

Youngsters Migration Database

Group ID - 44

Student ID and name:

Mohsin Pathan: 202412070

Hem Shah: 202412094



Date of submission: 13th November 2024

Table of Contents

Chapter No.	Chapter Name	Page No.
1	Software Requirements Specification (SRS):	4
	1.1 Problem Description	4
	1.2 Requirement Collection	10
	1.2.1 Background Reading	10
	1.2.2 Interview	35
	1.2.3 Questionnaire/Survey	38
	1.2.4 Observation	40
	1.3 Fact Finding Chart	41
1.4 Requirements List	42	
1.5 User Privileges	43	
2	Database Design:	46
	2.1 Noun Analysis	46
	2.2 Schema and ER diagram design	53
	2.3 ER Diagram Improvement	56
	2.3.1 Identify Entity Types	56
	2.3.2 Identify Relationship Types	57
	2.3.3 ER Diagram Analysis	59
	2.4 Mapping ER Model to Relational Model	61
2.5 Create DDL Scripts	63	

	2.5.1 Domain Constraints 2.5.2 Key Constraints	63 66
3	Normalization of Database: 3.1 Normalization and Schema Refinement 3.1.1 Original Design of the database 3.1.2 Dependency Analysis 3.2 Redundancies and anomalies Documentation 3.2.1 Redundancies 3.2.2 Anomalies 3.3 Normalization 3.3.1 1NF 3.3.2 2NF 3.3.3 3NF/BCNF	68 68 68 69 78 78 79 80 80 81 84
4	Implementation of Database: 4.1 Revised DDL Scripts 4.2 Database Population 4.3 SQL Queries	96 96 100 166
5	Interface Implementation: 5.1 Setup JDBCandBasic GUI 5.2 CRUD Operations in GUI	206 206 210
6	Technical Issues and Solution: 6.1 Technical Issues 6.2 Solutions	225 225 226

Chapter 1: Software Requirements Specification (SRS)

1.1 Problem Description

Migration is the process of moving from one region, place or locality to another in search of better opportunities to settle. Migration in India is not new and historical accounts show that people have moved in search of work, in response to environmental shocks and stresses, to escape religious persecution and political conflict. Youngsters migrate from one place to another for new opportunities and to explore more options which may include various factors like education, employment, marriage and many more. However, there is no comprehensive system to track this migration, making it difficult for policymakers and researchers to identify patterns, understand the reasons behind the movement, and address the challenges faced by young migrants. Without accurate data, government initiatives aimed at regional development, job creation, and infrastructure planning may not effectively meet the needs of young people. Keeping a track of these migrations is important as well as beneficial for the migrants as they will be informed about various opportunities in the area where they went. So after addressing the problem we started our research and reasoned the background of research papers to find the problem domain.

After going through certain research papers and studies we determined the scope of our project lied within India and domain we determined to work on were the various reasons why people migrate in India so we can keep track of the migrations that are happening in India, get the reason for migration and with the help of this data we can help the migrants to get the opportunities provided by Government and various schemes that they can get benefited from. The influence of government policies on migration. For this we read 3 research papers in which one was MAHA MTS which is the live database example of migration tracking system in Maharashtra. It helps us to gather information on how to create a certain system. The next research paper was related to IPUMS which is an international level portal that tracks migration all over the world. The next paper we studied was related to the e-shram portal which is the Indian system that helps us gather information about the migrants. After all the background reading of 10 research papers we were able to conclude the main reasons why people migrate from one place to another, they were:

- 1.Education migration
- 2.Employment migration
- 3.Marriage migration
- 4.Health issues related to migration
- 5.Climate change issues of migration
- 6.Lack of security issues in migration
- 7.Social security or exclusion issues in migration
- 8.Two database reviews of IPUMS as an international resource and the e-Shram portal
- 9.An existent study of the database Maha MTS

After we defined the problems related to our domain we conducted an interview with the company that wants to implement the database. We gathered information for the database and from the interview we got to know the need to create this database was to track migrant data as it was not proper, analyze the data and form a web application that helps the migrants to fulfill their purpose of migration. To gather more trends and patterns on data we created a questionnaire and surveyed various people all over India so that we can get an idea of what exactly they needed. We made an anonymous survey so that no one feels violated. We asked people for their age, gender, education level, occupation(if they had any), their place of origin, their current location, how long they have stayed there, how many times they have migrated and the reasons why they have migrated and the challenges they faced on migration that is any cultural difference that are present how they adapted to situations. On the basis of our survey we were able to get the outputs that we displayed in the SRS. On the basis of the outputs we gathered requirements that were:

- Age Range of the migrated people
- Gender of the migrated people
- Education level of the migrated people
- Occupation of the migrated people (optional)
- Place of origin
- Current location
- Duration at migrated location (patterns)
- Frequency of migration (patterns)
- Reasons for migration Challenges faced after migration

Observations that we made were:

Most of the people that have migrated away for any purpose are in the age group of 21-25, followed by 15-20 and then there are others that helps us to observe that mostly people migrate during their young ages and people are settled after the age of 26 so helps us find the age patterns of people migrating. The number of men and women that are migrating is in the ratio of 60-40 around according to the survey which helps us to make an observation that men migrate partially more than women do. The number of educated people is migrating more than the uneducated ones for new job opportunities, education or any other reason. On the basis of current location and place of origin we can observe the distance migrated by people in order to fulfill their wishes. From the survey we also get to know most people have migrated recently. We also are able to observe that many people have migrated only once to a new location. The most common reason for migration in youth has been education followed by employment and then by marriage and other factors. We can also observe the challenges faced by people like weather, home sickness, food, housing, cultural differences, language barrier, new people, health maintenance, pollution, marital issues and many more... Migration is draining talent from rural or underdeveloped regions. Migrants can be made aware of the government schemes and can be helped to sustain themselves in the alien surroundings.

After making all the observations starting from background reading, then defining the project problems, performing an interview to get a look towards the problems that we have faced whilst creating the same project earlier. Some references to the earlier work done, creating and formulating surveys, doing field work and observing we arrived to the

requirements for creating our database, however it is not complete and can have changes as we are going to create the database as an iterative model so any changes can be made during any part of the project. After all the observations we arrived at the requirements gathering for our project for now. They were:

1. Age-based migration patterns
2. Gender ratio analysis
3. Education level of the migrated people
4. Distance of migration (mostly used to find if the migration is intrastate or interstate)
5. Recent migration trends
6. Frequency of migration
7. Primary reasons of migration
8. Challenges faced by the migrants
9. Information on talent drain

Some of the other requirements we needed were the government policies information, health issues for people that have migrated, the climatic issues and all different kinds of issues that affect the migrants in one way or the other. Some of the functional and non-functional requirements for our project that we could determine so far are:

Functional Requirements:

1. User authentication
2. User authorization
3. Data collection (age, gender, distance, education, challenges, etc.)
4. Search options after creating the database
5. Government scheme integration
6. Survey integration

Non-functional Requirements:

1. Performance
2. Scalability
3. Security
4. Usability
5. Availability
6. Reliability

The youngsters migration database will be useful to various stakeholders such as Government officials and policymakers to track migration trends, design programs that help the migrants and deploy various schemes for them. Researchers to access data patterns on the migrants. NGOs and social organizations to identify vulnerable migrant groups and offer targeted assistance and finally help the migrants to get to the relevant schemes, services and employment opportunities.

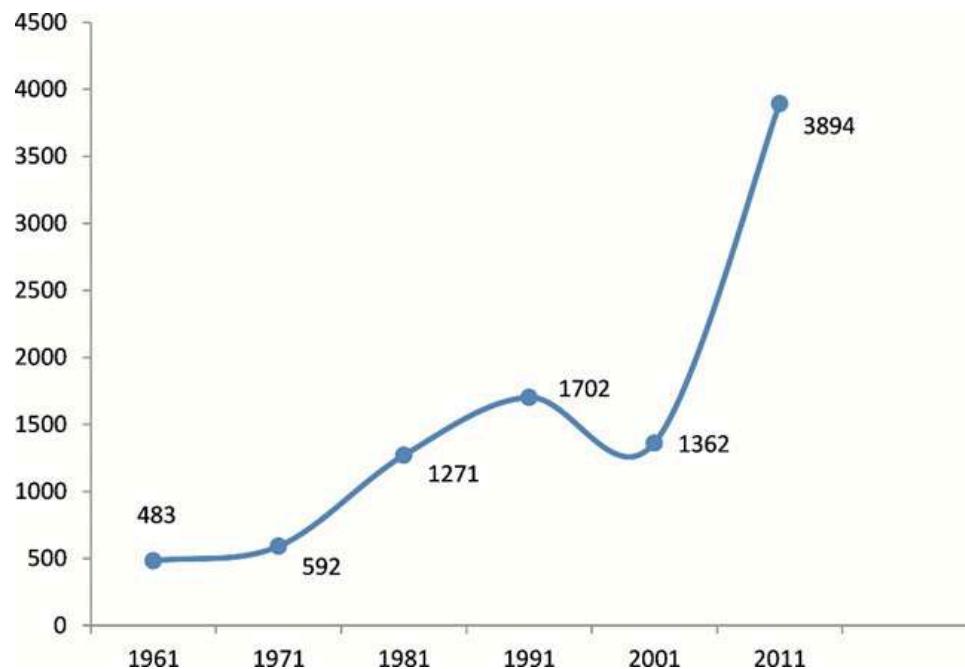
This database allows us to gather information on migrants and has various functions like data management which provides capabilities for administrators to manage, edit and delete migration records. Offers search functionalities on the basis of age, gender, education, etc so that we can target people on the basis of the scheme that is needed for them. Various government schemes related to the migrants can be sent to them and work can be done to help them provide opportunities in the area they are living. This concludes our brief description of the project we are working on.

1.2 Requirement Collection

1.2.1 Background Reading

BR-1 Changing livelihoods at India's rural–urban transition

- This paper describes the important facts of migration from rural areas to urban areas over the decades. It is divided into the shift of employment from agricultural sector, rapid growth at the bottom of the urban system with rural villages turning into urban and significant increases in labor migration
- According to the International Labor Organisation, agricultural employment 250 million jobs in 2004 to 215 million jobs in 2016.
- Urbanization of various rural villages into urban towns called Census Towns (CT) that meet for the first time the threefold Indian Census definition of urban.



-
- The relation of the first two points is that 75% of the CT became non-agricultural villages which referred to our first concluded point of shifting from agriculture with decades.
 - The focus of the paper shifts to the migration of people for labour activities with the terms circular migration and seasonal migration, it mainly focuses on one aspect or reason for migration that is the term ‘bread-winning’ or commonly called employment.
 - Circular migrations can be defined as people moving between two or more locations in a repetitive pattern, and often unified for taking seasonal advantage and overlaps with seasonal migration.
 - The paper research studies help us to find various reasons for migration like employment and also to figure from where and where are people moving as stated from rural areas to urban areas, this makes our data clear on that.

BR-2 The Changing Pattern of Internal Migration in India Issues and Challenges

- The paper has many references to the NSS (2007-08) data and discusses many aspects that are important for our domain of the project.
- The first point that it addresses is the trend of increasing migration of rural women is comparatively increasing than that of men. A reason supporting it is the success of the NREGA in village itself giving men more opportunities.

NSS rounds	Rural		Urban	
	Male	Female	Male	Female
38th (1983)	7.2	35.1	27	36.6
43 rd (1987/88)	7.4	39.8	26.8	39.6
49 th (1993)	6.5	40.1	23.9	38.2
55 th (1999/00)	6.9	42.6	25.8	41.8
64 th (2007/08)	5.42	47.3	25.9	45.62

Source: Authors Calculation from various NSS Rounds

- The second point it describes is the short-term migration which can be interpreted as inter-district or intra-state migration and the patterns discovered there too were gender specific to women. The percentage of women migrants is more than that of men in a smaller domain as well.

Table-2: Percentage distribution of migrants in different distance categories, NSS, 1999/00 & 2007/08 (Duration<5yr)

Types of migration	Total		Rural		Urban	
	M	F	M	F	M	F
2007/08						
Intra-district	37.59	59.05	52.5	69.57	27.71	38.32
Inter-district	34.71	30.33	27.77	24.15	39.31	42.51
Inter-state	26.27	10.33	17.77	6.07	31.9	18.72
International	1.43	0.29	1.95	0.21	1.08	0.45
1999/00						
Intra-district	47.78	63.09	59.84	71.98	37.77	43.47
Inter- district	30.94	26.64	23.06	21.18	37.47	38.67
Inter-state	19.72	9.94	15.08	6.53	23.57	17.46
International	1.56	0.34	2.01	0.31	1.19	0.4

Source: Author's Calculation from various NSS rounds

- The next point it talks about is the inter-state net migration in India which discusses the fact that inter-state migration is more among males and the mobility is from the states with higher levels of

poverty and illiteracy. Below is the table and fig for all the states' net migration.

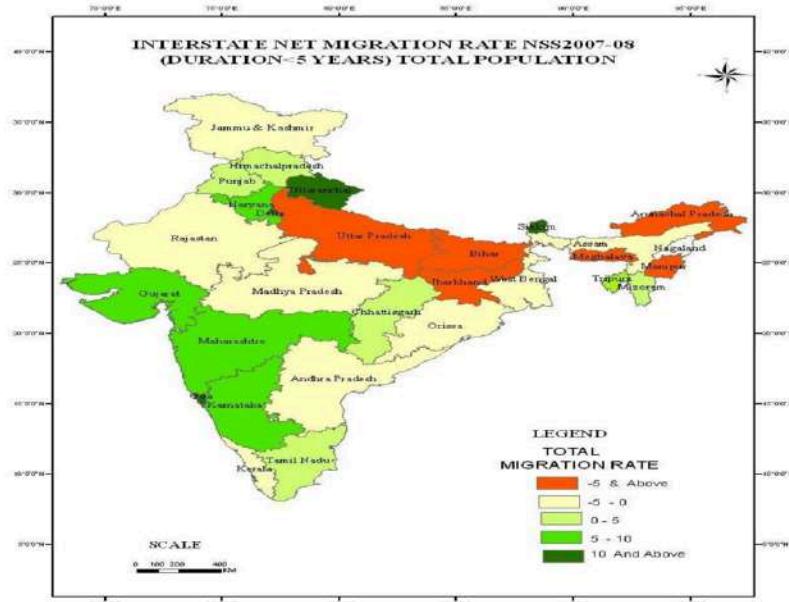


Table-4: Interstate net migration rate (Duration<5yr), NSS 1999/00 & 2007/08

State	2007/08			1999/00		
	M	F	T	M	F	T
Andhra	-2.31	-1.59	-1.95	0.91	-0.03	0.45
Assam	-2.28	-2.44	-2.36	-1.52	-1.13	-1.34
Bihar	-23.25	-12.3	-17.99	-13.77	-10.64	-12.27
Chhattisgarh	2.43	4.59	3.48			
Delhi	93.38	44.57	71.51	-18.74	-26.81	-22.42
Gujarat	13.77	5.19	9.72	2.88	3.47	3.17
Haryana	5.08	7.73	6.32	18.94	25.07	21.81
Jharkhand	-9.63	-5.98	-7.86			
Karnataka	12.84	5.73	9.33	-2.98	-1.42	-2.21
Kerala	-2.44	1.1	-0.58	0.69	-0.46	0.09
MP	-0.87	-1.65	-1.24	2.41	2.89	2.64
Maharashtra	13.51	6.1	9.94	12.56	10.44	11.54
Orissa	-6.71	-1.87	-4.27	0.3	-0.93	-0.31
Punjab	7.92	1.29	4.8	13.96	3.31	8.92
Rajasthan	-2.46	-0.16	-1.34	-3.4	-0.35	-1.93
Tamil Nadu	1.55	0	0.76	0.47	0.77	0.62
Uttar Pradesh	-11.59	-5.52	-8.66	-2.69	-2.18	-2.45
Uttarakhand	33.75	18.67	26.41			
West Bengal	-2.55	1.6	-0.53	1.4	3.45	2.4

Source: Author's Calculation from various NSS rounds

- Next it also talks about the employment patterns of migrants before and after migration, in short it was noted that there was an increase

in the per capita income as well as employment of the migrated workers.

Table-7: Percentage Distribution of Migrants by Occupation Before and After Migration (Dur <5yr, 15-59 age)

Activity Status	Male		Female	
	BM	AM	BM	AM
Total				
Self employed	28.86	26.01	38.67	41.96
Salaried/wage earner	40.06	54.41	12.8	19.17
Casual Laborer	31.07	19.59	48.53	38.88
Rural				
Self employed	26.88	33.33	40.24	47.04
Salaried/wage earner	32.81	35.23	6.85	9.01
Casual Laborer	40.31	31.43	52.91	43.95
Urban				
Self employed	30.09	22.26	31.88	21.85
Salaried/wage earner	44.55	64.22	38.44	59.34
Casual Laborer	25.37	13.52	29.68	18.81

Source: Author's Calculation from NSS 64th round(2007/08)

- This paper helps us to conclude two parts of our findings as the gender aspect of migration and the difference of migration from interstate to intrastate.

BR-3 Migration and health: A systematic review on health and health care of internal migrants in India

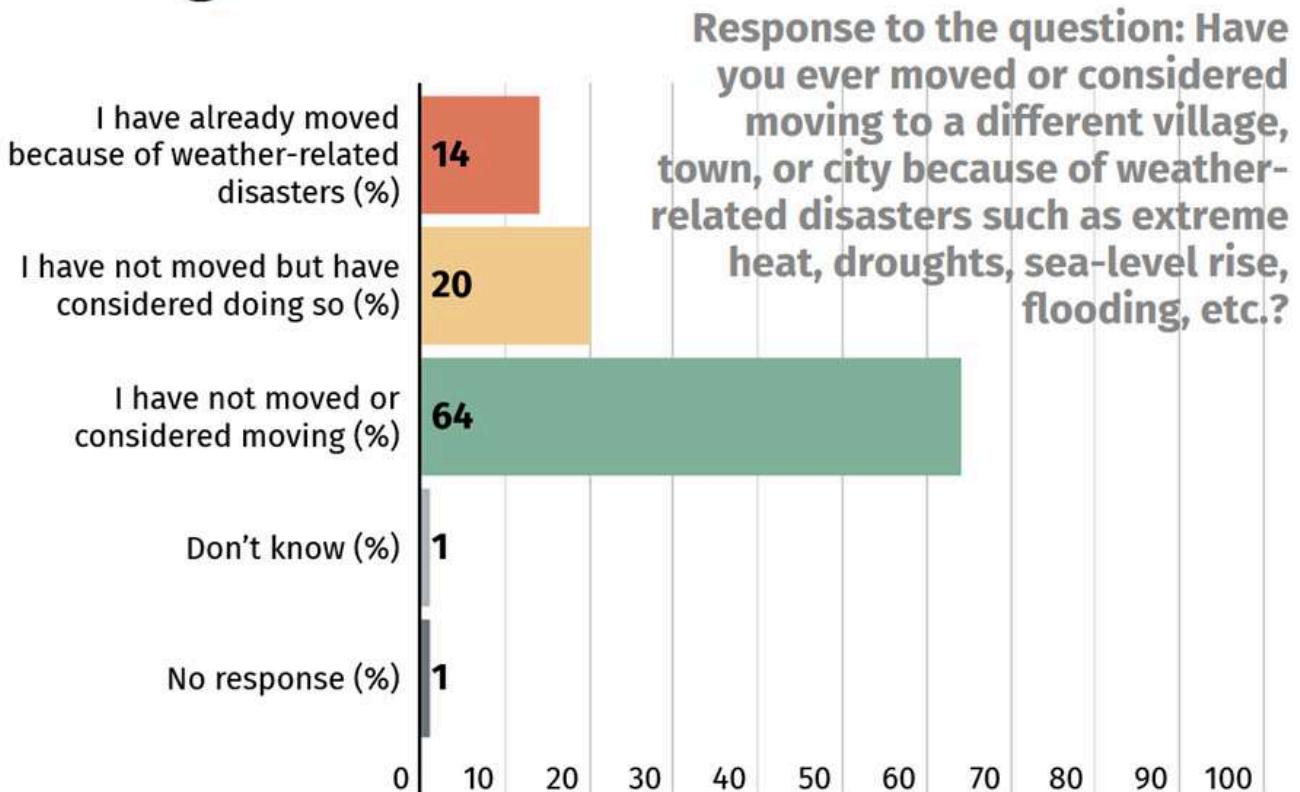
- This paper in general talks about the health issues of the migrants that have migrated from one place to another place.
- The poor migrants often are from the less-developed Indian states to the large and metropolis cities. They eventually get absorbed in the informal sector as daily wage laborers or as temporary salaried employees in the industries, factories, and construction companies.
- It discusses various diseases that are prevalent in the migrant workers and how many of them are related to blood pressure.

-
- Some common problems determined by the paper were hypertension, fasting blood glucose, higher body mass index, diabetes, and vascular diseases whereas their rural siblings did not have that kind of problems
 - It also discusses the reasons for their diseases like alcohol and tobacco.
 - Some other problems discussed in the paper were headache, fever, malaria and fatigue. There are many migrants that face the issue of acute respiratory diseases.
 - Other diseases were skin diseases, sunstroke, myalgia.
 - Some had accidents whereas some migrant women were subjected to sexual harassment.
 - The paper helps us conclude the problems of migrants related to the health issues and some of the cultural issues that need to be addressed to help them be more impactful towards their contribution in the economy.

BR-4 Climate change effects a major concern for Indians, 14% migrated due to weather-related disasters

- This article talks about the displacement of people from one place to another because of weather-related issues like floods, drought, water logging, eroded rivers, etc.
- Then it also talks about the fact that 14% of people in the affected areas migrated because of these prevalent issues.

1 in 3 Indians Have Moved Or Considered Moving Due To Weather Related Disasters



Source: [Climate Change in the Indian Mind, 2023](#)



- Various causes of climatic migration include agricultural pests and diseases, extinction of plant and animal species, severe heat waves, droughts and water shortages, severe air pollution, famines and food shortages, severe cyclones and severe floods

BR-5 Marriage Migration in India

- This paper talks about the largest permanent migration that happens and is called as marriage migration in which the bride migrates from one region to another because of marital reasons.
- The first important point is the bifurcation of marriage migration in northern and southern states, as the amount of migration is more in northern states which relates to the determination of irregular sex ratio resulting into imbalance of men and women in a particular area. Whereas the southern states are less as mostly they marry in same areas and cousin marriage is allowed. This results into a total balanced out state in the southern states.
- The next important point focusses on average marriage migration and gross marriage migration. It is very low for average marriage migration as women moving out of one place are cancelled out by the women moving in to the same place. However, the gross marriage migration is the most in case of migration in India and has resulted as a permanent migration in India.
- Its next focusses on the facts of spatial variance of women from one area to another which is an important data collection for our project. The number of women moving out between 16-22 is the most and it declines with the age of 40. Many of the earlier migrated women are either half or one hour away from their natal home so the distance of women migrated also is an important aspect of it.
- The paper also discusses various problems for brides in their daily household life and how they are mistreated on the basis of low dowry or many other reasons.

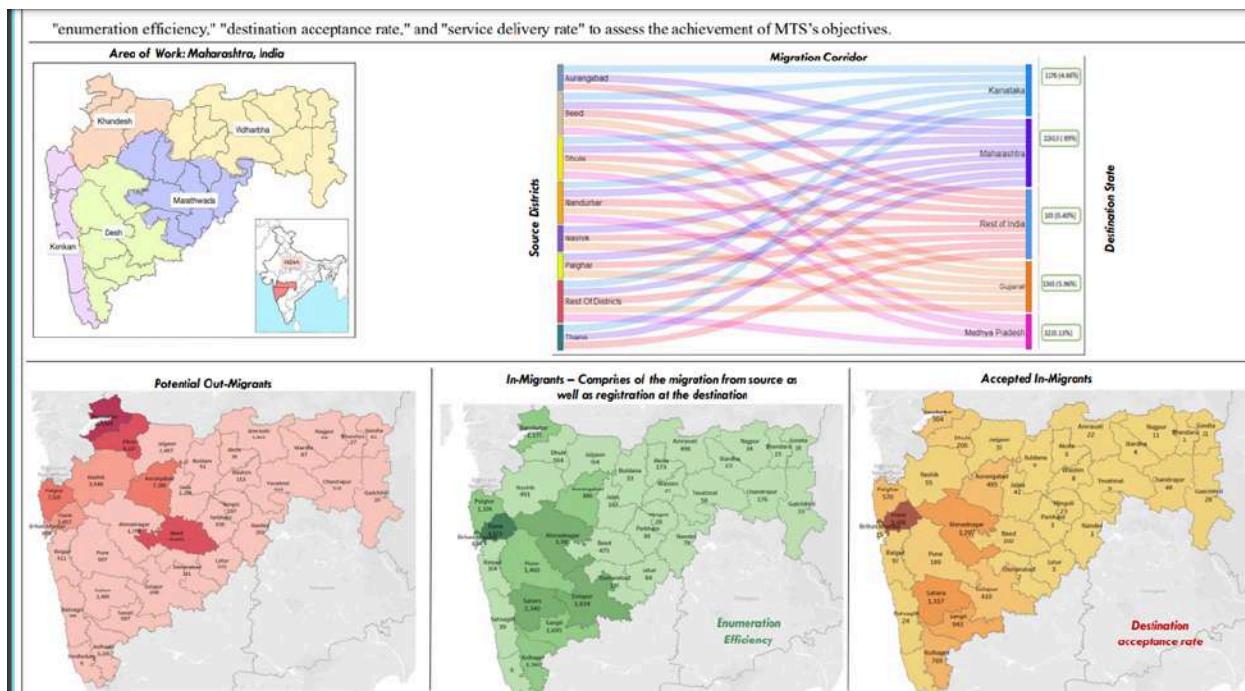
Table 1: Marriage migration, female autonomy, and marriage customs

	All India	Northern states	Rest of India		
	Rural	Urban	Rural	Urban	
Fraction women over 22	1.00	0.29	0.07	0.43	0.21
Fraction migrate for marriage	0.66	0.83	0.68	0.62	0.39
Fraction women over 22 illiterate	0.57	0.71	0.43	0.56	0.27
Fraction women migrate illiterate	0.52	0.71	0.39	0.54	0.27
Fraction do any non-domestic work	0.30	0.34	0.14	0.36	0.18
Fraction with any work outside home	0.07	0.07	0.05	0.07	0.09
Hours to natal home on marriage	3.42	3.48	4.81	2.91	3.87
Who chose your husband?					
Respondent herself	0.05	0.02	0.03	0.07	0.06
Respondent and parents	0.34	0.22	0.30	0.38	0.43
Parents alone	0.60	0.75	0.66	0.54	0.50
Other	0.01	0.00	0.00	0.01	0.01
How long had you known your husband before you married him?					
On wedding/gauna day	0.68	0.87	0.85	0.59	0.54
Less than a month	0.09	0.03	0.04	0.13	0.14
Less than a year	0.11	0.04	0.03	0.13	0.18
More than a year	0.04	0.03	0.04	0.04	0.06
Since childhood	0.08	0.03	0.05	0.11	0.08
In your community (jati) do people:					
Marry a daughter in her natal village?	0.48	0.30	0.48	0.56	0.57
Marry a daughter to her cousin?	0.38	0.16	0.25	0.49	0.47
Do you need permission to visit the health center?					
Yes	0.73	0.83	0.76	0.73	0.61
If yes, can you go alone?	0.66	0.44	0.66	0.74	0.83
Do you need permission to visit the home of relatives or friends in the neighborhood?					
Yes	0.73	0.79	0.73	0.72	0.69
If yes, can you go alone?	0.69	0.53	0.67	0.74	0.79
In your community is it usual for husbands to beat their wives if:					
She goes out without telling him?	0.39	0.50	0.32	0.38	0.29
Her natal family does not give expected gifts?	0.29	0.34	0.26	0.29	0.22
She neglects the house or children?	0.35	0.37	0.26	0.37	0.29
She doesn't cook food properly?	0.29	0.34	0.21	0.31	0.23

BR-6 Mitigating Vulnerabilities of Seasonal Migration Case of Maharashtra Migration Tracking System

- This paper discusses the web application launched by the Maharashtra Government known as the Maha MTS, and it is the same model as we are working on which allows people to apply on the web portal and find various opportunities in the area they have migrated to.

-
- The aims of Maha MTS are:
 - i. Enumerate seasonal migrants in high burden districts
 - ii. Deliver essential social services and entitlements to migrant families
 - iii. Develop technology platform for seamless service transfer
 - iv. Leverages existing in government field functionaries and officers for uninterrupted transfer and access to state's social protection services
 - This study critically assesses the operational feasibility of MTS in mitigating health and nutrition vulnerabilities among migrant populations.
 - Analysis of data for 37,848 seasonal migrants (for the year 2021-22) from high-burden districts in Maharashtra was conducted, employing operational efficiency indicators like "enumeration efficiency," "destination acceptance rate," and "service delivery rate" to assess the achievement of MTS's objectives.
 - The Maha MTS mobile-based application has proven instrumental in enumerating a total of 69,195 potential migrant beneficiaries, with 21,472 individuals actually undergoing migration. Among these migrants, 10,161 were successfully accepted at their intended destinations, allowing services to be delivered to 31% of the beneficiaries.
 - Most seasonal migrants were 'Intra-State' migrants (89%), about 6% went to Gujrat, 4.6% Karnataka and the rest to other states.



BR-7 Internal Migration for Education and Employment among Youth in India

- States with low literacy rate such as UP, Bihar, Rajasthan face challenges while providing higher quality education which leads to higher fertility rates and an outflow of human capital

Finding: We can get the states from where young people move out

- Intra state migration with 45% is on a higher side than inter-state migration that accounts to only 17% of the total migration

Finding: Intra state migration is more than inter-state migration

- The flow of human capital across states has significant implications for balanced regional development in India like Gujarat, Maharashtra benefit from this influx while other regions suffer from losing educated individuals.

Finding: Migration effects on various states in India

There are four migration streams: rural-rural, rural-urban, urban-rural and urban-urban. Further, the stream can be intra-district, intra-state and inter-state.

Table 1: Net Attendance Ratio by Broad Class Group (All India)

Class Group	Rural			Urban			Rural +Urban		
	Female	Male	Total	Female	Male	Total	Female	Male	Total
I-V	83	86	84	84	86	85	83	86	84
VI-VIII	54	59	57	64	67	65	56	61	59
IX-X	35	40	38	51	52	51	39	43	41
XI-XII (general education)	19	25	22	39	39	39	25	29	27
XI-XII *(all education)	20	25	23	39	40	40	25	29	27
Post higher secondary (general education)	5	8	6	14	13	14	7	9	8
Post higher secondary (all education.)	6	10	8	21	20	21	10	13	12

* includes diploma with minimum requirements below higher secondary

Education is categorized in three classes in the survey: (i) general education, (ii) technical and professional education and (iii) vocational education. All education includes (i) (ii) and (iii)

Source: Government of India (2010a)

Table 2: Distribution of internal migrants by last usual place of residence for each component of rural-urban migration streams

Migration streams	Intra district	Inter district	Intrastate	Interstate	All
			(Intra district+ Inter district)		(intrastate+ Interstate)
55th round (1999-2000)					
Rural-to-rural	75.3	20.1	95.4	4.6	100
Rural-to-urban	43.8	36.5	80.3	19.6	100
Urban-to-rural	46.5	33.5	80.0	20.0	100
Urban-to-urban	36.6	43.5	80.1	19.9	100
64th round (2007-08)					
Rural-to-rural	72.4	23.2	95.6	4.4	100
Rural-to-urban	41.2	33.6	74.8	25.2	100
Urban-to-rural	48.8	33.8	82.6	17.5	100
Urban-to-urban	27.9	49.2	77.1	22.9	100

Source: NSSO (2010) Report on Migration in India

Table 3 : Number of Migrants by Reason for Migration (15-32 years)

Reason for migration	Migration Streams				Total
	Rural-Rural	Rural-Urban	Urban-Rural	Urban-Urban	
In search of employment	1,810,512	5,707,409	590,054	2,353,658	10,461,633
Education	708,610	1,617,152	604,671	915,401	3,845,834
Marriage	60,048,081	8,070,261	3,812,910	5,619,806	77,551,058
With parent/earning member of family	3,267,400	5,482,397	883,167	4,354,838	13,987,802
Others	2,120,095	686,414	648,865	758,396	4,213,770
Total	67,954,698	21,563,633	6,539,667	14,002,099	110,060,097

10

This paper helps us to determine the different field levels in different states according to the NSSO websites within the age group of 15-32 which exactly lies in the domain of our project and helps us determine the people we are going to figure out the ways to manage them in a database.

**Table 6: Share of migrant population by states and educational attainment in last 10 years
(Age group 15-32 years)**

Destination States	Education level						Total
	Illiterate	Below Primary	Primary/ Middle	Secondary	Higher Secondary Diploma	Graduate and above	
Jammu & Kashmir	0.3	0	0.3	0.1	0.2	0.1	0.2
Himachal Pradesh	0.9	0.5	0.8	0.3	0.4	0.6	0.6
Punjab	7.8	10	5.6	4.2	3.9	3.9	5.7
Uttaranchal	3.6	2.3	2.7	2.8	2.3	2.4	2.8
Haryana	7	5.9	6	7.4	5.5	5.4	6.3
Delhi	14.1	10.8	17.1	19.4	14.8	15.8	16
Rajasthan	9.3	7.4	4.3	3	5.2	3	5.3
Uttar Pradesh	13	11.5	4.9	7.9	6.3	11.1	8.4
Bihar	2.7	3.1	1.1	1.6	1.1	1.1	1.6
NE States	0.4	0.8	0.7	0.3	0.3	0.4	0.5
Assam	0.2	0.2	0.3	0.3	0	0.1	0.2
West Bengal	6.1	3.6	3.8	3	0.8	3.9	3.8
Jharkhand	0.6	1.9	0.8	0.9	0.3	0.7	0.8
Orissa	1.1	2.4	1.2	1	1.5	1	1.2
Chhattisgarh	1.5	1.3	1.8	0.7	1.5	0.6	1.4
Madhya Pradesh	4.4	5.4	3.1	2.1	2.4	2.4	3.2
Gujarat	5.2	9.9	11.5	7.9	4.9	4	7.8
UTs except Delhi							
Delhi	1.8	1.8	1.7	2.9	4.7	3.5	2.5
Maharashtra	12.6	11.8	19.5	19.5	16.9	14.7	16.6
Andhra Pradesh	2.8	3	2.7	3.4	3.3	3	3
Karnataka	3.1	2.7	5	5.7	14.9	14.1	6.9
Goa	0.2	1.4	0.5	0.9	1	1.5	0.8
Kerala	0.4	0.6	1.8	1.3	2.3	2	1.4
Tamil Nadu	1.1	1.6	3	3.3	5.3	4.6	3.1
Total	100	100	100	100	100	100	100

- This table gives the share of each state's contribution in migrants education attainment in the last 10 years
- This table helps us to bifurcate the migrants of different states in India

Table 9: Transition across Broad industry groups after migration (age group 15-32)

	Agriculture and mining	Manufacturing	Construction	Trade and Hotels	Transport	Other Services
Agriculture and Mining	50.9	7.3	2.7	2.7	1.6	2.0
Manufacturing	0.7	8.0	0.2	0.4	0.2	0.3
Construction	0.8	0.7	3.7	0.2	0.3	0.1
Trade and Hotels	0.3	0.7	0.1	5.1	0.1	0.5
Transport	0.1	0.2	0.0	0.1	2.1	0.1
Other Services	0.2	0.2	0.1	0.1	0.1	6.9

**Table 7: Migration for employment by current state and location of last usual place of residence
(Age group 15-32 years)**

State	Last Usual Place of Residence									Total	
	Same District		Other District		Same State		Other State				
	Rural	Urban	Rural	Urban	Rural	Urban	Rural	urban			
Jammu & Kashmir	11.7	20.8	15.8	17.2	27.5	38	28.8	5.8	100		
Himachal Pradesh	30.5	8.2	18.2	6.9	48.7	15.1	21.7	14.5	100		
Punjab	6.1	0.9	5.8	3.4	11.9	4.3	75.1	8.6	100		
Uttaranchal	11.5	3.6	9.6	8.9	21.1	12.5	46.8	19.7	100		
Haryana	4.1	3.5	12.2	3.3	16.3	6.8	56.3	20.5	100		
Delhi	0.6	1	0.9	2.9	1.5	3.9	73	21.6	100		
Rajasthan	43.3	3.4	19.3	9.1	62.6	12.5	16.1	8.8	100		
Uttar Pradesh	12.6	6.7	34.6	18.6	47.2	25.3	15.5	12	100		
Bihar	33.6	4	34.8	7.6	68.4	11.6	8.5	11.5	100		
NE States	26.2	7.3	15.2	13.4	41.4	20.7	28.8	9.1	100		
Assam	29.4	2.3	42.1	14.6	71.5	16.9	9	2.7	100		
West Bengal	20.5	4.8	19.4	12.1	39.9	16.9	32.3	10.8	100		
Jharkhand	33.9	1.4	47.1	5.1	81	6.5	2.5	10	100		
Orissa	30.4	4.9	40.5	8	70.9	12.9	10.4	5.8	100		
Chattisgarh	38.4	2.7	14.4	12.2	52.8	14.9	13.5	18.8	100		
Madhya Pradesh	24.7	13.5	24.9	12.5	49.6	26	17.5	6.8	100		
Gujarat	13.8	3.5	9.3	7.2	23.1	10.7	61.3	4.8	100		
UTs except Delhi	5.3	2.1	1.3	0.9	6.6	3	68.3	22.1	100		
Maharashtra	11.3	4.4	19.8	14.6	31.1	19	42.3	7.6	100		
Andhra Pradesh	37	3.4	28.5	20.2	65.5	23.6	5	5.9	100		
Karnataka	11.4	2.2	26.2	23	37.6	25.2	18.3	18.9	100		
Goa	3.4	0.4	0	1.6	3.4	2	53	41.6	100		
Kerala	22.4	2	40.2	14.5	62.6	16.5	15.1	5.9	100		
Tamil Nadu	14.8	9.5	29.2	27.5	44	37	10.6	8.5	100		
Total	17.1	4.1	19.9	13.3	37	17.4	34.9	10.7	100		

-
- This is the table for migrant employment in different states which helps us to bifurcate the people with employment on the basis of different states.

BR-8 Data Resource Profile: IPUMS-International

- The Integrated Public Use Microdata Series-International (IPUMS-International) is a global initiative to disseminate census microdata. Since its inception in 1999, IPUMS-International has partnered with over 100 national statistical agencies to provide standardized census data from more than 82 countries, making it the largest collection of publicly available census microdata in the world.
- IPUMS-International currently offers 277 anonymized microdata samples, spanning from 1960 to 2011 and covering over 600 million persons globally. These data cover a broad range of population characteristics, including fertility, migration, education, and labor force participation. The data are available free of charge through an online system, where researchers can create customized datasets tailored to their specific needs.
- The data series uses a uniform coding system, allowing variables to be harmonized across countries and time periods without losing detail. IPUMS-International offers advanced features like constructed household relationship variables and spatiotemporal harmonization of geographical boundaries to enable precise research and comparison across regions.
- The database includes detailed geographical data, often down to the second administrative level (e.g., municipalities or districts), making it suitable for analyzing regional variations in public health, economic development, and migration patterns. IPUMS

also provides GIS boundary files to support mapping and spatial analysis.

- The strengths of IPUMS-International include its global reach, harmonized data, and free access to vast census datasets. The main limitation is that the data are cross-sectional, meaning individuals cannot be tracked across censuses. Despite this, the large sample sizes and harmonized variables provide precise analyses across time and space.
- This paper helps us to figure the database creation of the whole world migration as an idea

BR-9 Visibilising Invisible Population: e-Shram Portal and Essentiality of Internal Migration Database in India

- This paper talks about various issues of internal migration various factors that affect them like the structural changes in the economy, regional inequalities, agrarian distress and urbanization that led millions of people to migrate
- By 2020, an estimated 600 million people were internal migrants in India, most of whom work in informal urban sectors.
- This paper talks about the lack of social security for the migrants. Internal migrants face exclusion, marginalization, and lack of social security in destination cities. They are often employed in informal jobs such as construction, street vending, and domestic work, yet remain invisible in policy frameworks. The COVID-19 pandemic exacerbated their vulnerability, exposing the lack of support and protection for this population.
- It talks about the need for social security of the migrants to protect them from extortion. The paper emphasizes the necessity of

registering migrants, which allows them to access entitlements and government benefits.

- The focus then shifts from it to the creation of e-Shram portal which is a comprehensive database of unorganized workers. The portal aims to register unorganized workers and provide them with access to social security benefits. By May 2023, over 289 million workers had registered, with more women than men, highlighting the importance of gender-specific protection measures.
- The registration process faces hurdles, such as the need for Aadhaar-linked mobile numbers and bank accounts, which many migrants do not have. Technical issues, confusion in job categorization, and regional disparities in registration also hinder the process. Misconceptions about the portal among workers further complicate effective registration.
- To ensure the inclusion of migrants in policy and development, the paper calls for a data-driven, rights-based approach. It recommends disaggregating and triangulating migration data to better understand the specific needs of different migrant groups, especially vulnerable populations like women and seasonal workers. The paper also stresses the need for collaboration between various stakeholders to improve social security provision for internal migrants.

BR-10 A Study on Social Security and Health Rights of Migrant Workers in India

- This paper is useful to gather information about the social security of Indian migrant workers.
- Social security consists of policies and programmes designed to reduce poverty and vulnerability.

- It can include social insurance, social assistance and other interventions.
- Social insurance programmes require regular contributions and provide insurance against shocks.
- Social assistance programmes are generally targeted at the poor and non-contributory. Interstate migrant workers, by and large, are poor while some of them are in the most vulnerable and in worst condition who need support through various social assistance programmes. In fact, we cover their eligibility for participating in social protection programmes.
- It is pertinent to identify the barriers to such participation and to examine what has been or can be done to minimize risks of exclusion and promote adequate coverage for interstate migrant workers.

Figure: 1.2
Components of Social Security or Social Protection

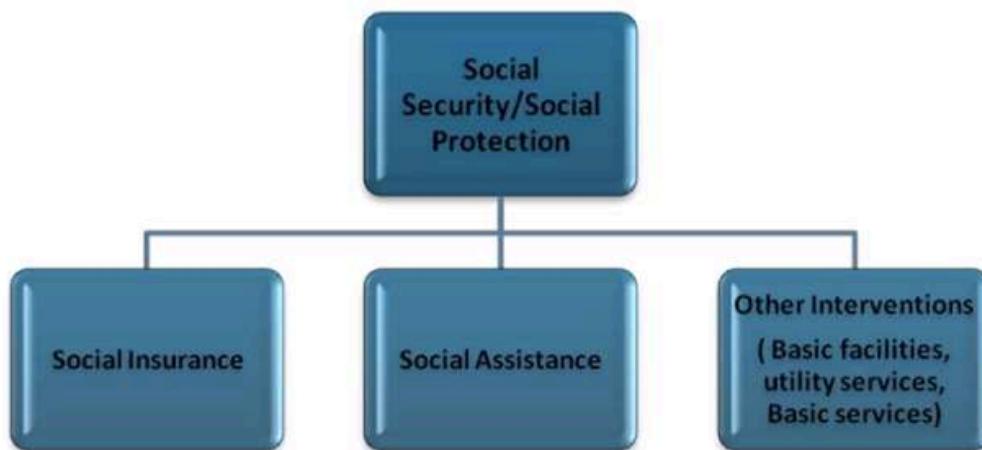


Table 2.3
Occupation of Interstate Migrant Workers Covered by Survey

Occupation of ISMW covered	ISMW in Delhi		ISMW in Gujarat		ISMW in Haryana		ISMW in Maharashtra	
	Number	%	Number	%	Number	%	Number	%
Construction workers	60	15	84	21	88	22	92	23
Factory workers in organized sectors such as Textiles and Chemicals	28	7	40	10	28	7	28	7
Factory workers in unorganized sector (include Artisans, workers / welding units, automobile workshops/ other workshops)	32	8	44	11	16	4	24	6
Trading activities such as hotel, restaurant, shops, wholesale and retail markets and street vending	68	17	44	11	60	15	60	15
Agriculture workers such as farming operations, sugarcane cutters, salt pans, cotton ginning)	4	1	16	4	20	5	12	3
Jewelry works, brick kilns, salt making, stone quarrying, Diamond cutting	4	1	28	7	8	2	8	2
Work with security agency as security guards	48	12	48	12	40	10	52	13
Rickshaw pullers, E – Rickshaw drivers, rag pickers	44	11	20	5	8	2	20	5
Domestic Workers	44	11	40	10	56	14	48	12
Daily wage workers with Television and film industry and e-commerce companies including delivery boys	20	5	16	4	12	3	28	7
Other Daily wage workers such as Plumbers, electricians and painters)	48	12	20	5	64	16	28	7
Total	400	100	400	100	400	100	400	100

Source: Primary Data

Table 2.4
Interstate Migrant Workers: Period of Stay in Host States

Duration of stay in host States	Delhi		Gujarat		Haryana		Maharashtra	
	Number of Responses	%						
Up to 3 months	12	3	22	5.5	10	2.5	21	5.25
Up to 6 months	15	3.75	30	7.5	40	10	27	6.75
Up to 9 months	19	4.75	90	22.5	65	16.25	80	20
Up to 1 year	21	5.25	60	15	82	20.5	67	16.75
2-3 years	98	24.5	85	21.25	99	24.75	97	24.25
3-4 years	152	38	72	18	65	16.25	92	23
4-5 years	63	15.75	30	7.5	21	5.25	10	2.5
Above 5 years	20	5	11	2.75	18	4.5	6	1.5
Total	400	100	400	100	400	100	400	100

Source: Primary Data

- There are several forms of human rights violations against interstate migrant workers. These are poor access to health services, poor access to social protection, poor access to education services, poor access to housing and sanitation, poor access to food, water and other utility services. There is clear evidence of poor access to health services for migrant workers in all four States.

Fig 2.3
Areas of Human Rights Violations



Table 2.5
Social Security and Health of Interstate Migrant Workers and Human Rights Violations:
State wise

Types of Violations	Delhi		Gujarat		Haryana		Maharashtra	
	Number of Responses	%						
Absence of proper accommodation/ Poor quality of accommodation	336	84	324	74	324	81	356	89
Inadequate safeguards / high risk at worksites	289	72.3	292	73	304	76	308	77
Non-Provisioning of entitlements of government schemes	248	62	260	65	244	61	276	69
Poor access to available schemes and services due to the lack of information and language barriers	204	51.2	212	53	224	56	220	55
Distress from ill treatment of local laborer	100	25	108	27	196	49	210	52.5
Long working hours	192	48	208	52	212	53	218	54.5
Absence of leisure time/ entertainment	156	39	152	38.2	162	40.5	170	42.5
Poor social interactions and lack of integration with the local community	154	38.5	156	39	158	39.5	206	51.5
Limited access to health care services	68	17	180	45	116	29	220	55
Distress from ill treatment and discrimination of employer	48	12	128	32	72	18	156	39
Exploitation of migrant labourers by middleman/ agents and the resultant low wage	48	12	76	19	54	13.5	174	43.5

Source: Primary Data

Note (1) Multiple Response Questions

Note (2): A significant number of ISMW are unaware of their problems, especially the long working hours and lack of leisure or entertainment. ISMW, mostly young and without family members in host States prefer to work overtime and earn maximum amount of salary

- There are many different discriminations against the migrant worker due to difference in regional practices that makes them left out and groupism is created and looked down on.

Table 2.6
Interstate Migrant Workers: Forms of Prejudices and Discriminations

Forms of Prejudices/ Discriminations	Responses of ISMW in Delhi		Responses of ISMW in Gujarat		Responses of ISMW in Haryana		Responses of ISMW in Maharashtra	
	Number	%	Number of Responses	%	Number of Responses	%	Number of Responses	%
Local people believe migrant labourers are polluting the environment by dumping wastes in public places. Migrant labourers are accused of dumping food and other wastes into common land and rivers/canals thus polluting the environment	374	93.5	360	90	348	87	356	89
Local people consider migrant laborers as outsiders and do not treat them as equals in the society. Social exclusion of ISMW a reality	370	92.5	348	87	344	86	362	90.5
Employers discriminate ISMW in the labour market in respect of wages and accommodation. A significant number of ISMW do not have much complaint about the wage discrimination as the current wage is quite attractive to them compared to the home State	228	57	260	65	236	59	276	69

Source: Primary Data

Note: (i). Responses of three categories of responders are included : A total of 2800 respondents –ISMW, Local Labourers and Employers have indicated various forms of prejudices and discriminations. These are presented under five broad heads

(ii).Multiple Response Questions and the total number of respondents are 2800

- This study is mainly limited to the worker conditions in 4 states of Gujarat, Haryana, Delhi, and Maharashtra but the main outcome we needed from this paper was about the social security concerns of the migrants.

Conclusion

After reading all the 10 research papers we have figured out the domains of our project and where we can gather data revolve around and briefing of the project description helps us find the points we need to work during this project:

1. Education migration
2. Employment migration
3. Marriage migration
4. Health issues related to migration
5. Climate change issues of migration
6. Lack of security issues in migration
7. Social security or exclusion issues in migration
8. Two database reviews of IPUMS as an international resource and the e-Shram portal
9. An existent study of the database Maha MTS

These points conclude the project description on background reading.

References

- BR-1 <https://www.sciencedirect.com/science/article/pii/S0305750X21002321>
- BR-2 <https://www.shram.org/uploadFiles/20130128045030.pdf>
- BR-3 https://www.researchgate.net/publication/326830056_Migration_and_health_A_systematic_review_on_health_and_health_care_of_internal_migrants_in_India
- BR-4 <https://scroll.in/article/1069564/climate-change-effects-a-major-concern-for-indians-14-migrated-due-to-weather-related-disasters>
- BR-5 <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=42b4648b6f9bdcbf834e35b37f917aeab2aae83a#:~:text=Marriage%20migration%20in%20India%20is,from%20their%20place%20of%20birth.>
- BR-6 https://cartography.genevahealthforum.com/images/posters/posters_2024_pdf/sankhe_tiksha.pdf
- BR-7 <https://www.shram.org/uploadFiles/20140122121634.pdf>
- BR-8 <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5837405/>
- BR-9 <https://journals.sagepub.com/doi/full/10.1177/23944811231174827#:~:text>To%20create%20a%20credible%20database,a%20comprehensive%20and%20challenging%20process>
- BR-10 https://nhrc.nic.in/sites/default/files/Approved_Health%20and%20social%20security%20ISMW_KDS-NHRC.pdf

1.2.2 Interviews

MoveForward Database: Interview Plan

System: MigrateMatrix

Project Reference: SF/SJ/2024/09

Participants: Hem Shah (MigrateMatrix)

Mohsin Pathan (MoveForward Database)

Date: 03/09/2024

Time: 14:30

Duration: 30 minutes

Place: Hem's Office

Purpose of interview:

Preliminary meeting to identify problems and requirements regarding youngsters' migration database for MigrateMatrix.

Agenda:

- Problems regarding the insufficient data available on migrant information
- The information regarding cultural differences and discrimination
- Problems regarding their lifestyle and financial conditions
- Initial ideas
- Follow-up actions

Documents to be brought to the interview:

- Rough plan of the data to be collected
- Any Government ID and documents related to current database system

Interview Questions

- 1) What are you going to do as a company with this data?
- 2) Why do you want to build this database as a company?
- 3) What is the current status of the datasets?
- 4) What is the main problem that you have faced in the past that hampered the progress of making this database?
- 5) What are the social problems you think that the migrants are facing?
- 6) Which is the most common problem that needs to be addressed for the migrants?
- 7) What will be the impact on society and the policies of the Indian Government for migrants by creating this database?
- 8) Why do young people migrate? How does it affect the rural areas and what is the data you have about distribution across regions?

MoveForward Database: Interview Summary

System: MigrateMatrix

Project Reference: SJ/SF/2024/09

Participants: Hem Shah (MigrateMatrix)

Mohsin Pathan (MoveForward Database)

Date: 03/09/2024

Time: 14:30

Duration: 30 minutes

Place: Hem's Office

Purpose of interview:

Preliminary meeting to identify problems and requirements regarding youngsters' migration database for MigrateMatrix.

1. Young migrants often face a mismatch between their qualifications and available job opportunities, but there's insufficient data to analyse this issue comprehensively. (*action: introduce Government initiatives and collaborate them area-wise to help the migrating people*)
2. Many young people move without being formally recorded leading to gaps in available data.
3. Existing datasets often don't distinguish between different age groups or reasons for migration, making it difficult to focus on youth. (*action: create a database that helps extinguish people on the basis of age*)
4. Don't have adequate data on people's reasons for moving that helps them. (*action: circulate the questionnaire on youngsters' migration database as much as possible*)
5. There are many cultural differences and discrimination between the migrants and locations. (*action: look for common values between them and look to address them*)
6. The living and financial conditions of people that have migrated.
7. Without data on Youth migration patterns, policymakers struggle to develop targeted problems that address the specific needs of youth migrants
8. Many young people migrate for better opportunities, causing a brain drain in rural or less developed areas, but there's limited data on how migration is affecting talent distribution across regions.
9. Further discussions need when more information is available. (*arrange a follow-up meeting with Hem (in about a week's time)*)

1.2.3 Questionnaires/Surveys

MigrateMatrix - Youngster's Migration in India -Survey

1. What is your age? 15-20 / 21-25 / 26-30 / Above 30
 2. What is your Gender? Male / Female / Prefer not to say
 3. What is your education level? 10th / 12th/ Graduate / Post-Graduate / other
 4. What is your occupation?
-

5. What is your place of origin?
-

6. What is your current location?
-

7. How long have you been staying at the migrated location?

Less than a year / Less than 5 years / More than 5 years / other

8. How many times have you migrated to a different location within 5 years?

Once / Twice / Thrice / other

9. What is the reason for your migration?

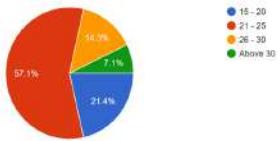
Education / Employment / Health / Social Exclusion / Lack of Security / Climate conditions / Marriage / Other

10. What are the challenges you faced after migration?
-

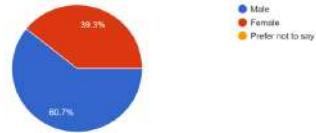
11. Suggestion and Feedback
-

Google form Responses

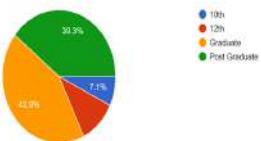
What is your Age ?
28 responses



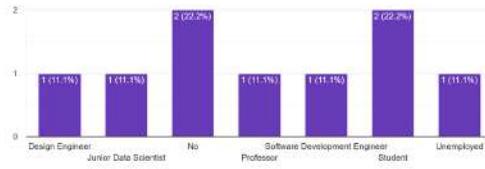
What is your Gender
28 responses



What is your Education Level
21 responses



Occupation if (If Any) ?
9 responses



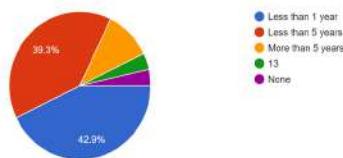
What is your Place of Origin ?
28 responses



What is your Current Location ?
28 responses



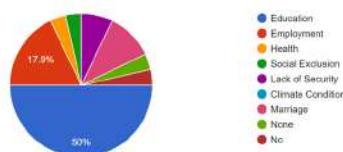
How long have you been staying at the migrated Location ?
28 responses



How many times have you migrated to a different location within 5 years ?
28 responses



Reason For Migration ?
28 responses



What are the challenges you faced after migration ?
28 responses

- Bleeding in multiculturalism
- Accommodation, Part time, Food
- Cultural differences
- No migration
- To find accommodations
- Transport and time management
- Cannot adjust with the environment
- None
- Language barrier

Suggestion and Feedback
6 responses

Please come only for masters or PHD and be prepared with finances

Good form

None

No

Not at all

You should complete your schooling from one place.

1.2.4 Observation

List of Observations:

- Most of the people that have migrated away for any purpose are in the age group of 21-25, followed by 15-20 and then there are others that helps us to observe that mostly people migrate during their young ages and people are settled after the age of 26 so helps us find the age patterns of people migrating.
- The number of men and women that are migrating is in the ratio of 60-40 around according to the survey which helps us to make an observation that men migrate partially more than women do.
- The number of educated people is migrating more than the uneducated ones for new job opportunities, education or any other reason.
- On the basis of current location and place of origin we can observe the distance migrated by people in order to fulfill their wishes.
- From the survey we also get to know most people have migrated recently
- We also are able to observe that many people have migrated only once to a new location
- The most common reason for migration in youth has been education followed by employment and then by marriage and other factors.
- We can also observe the challenges faced by people like weather, home sickness, food, housing, cultural differences, language barrier, new people, health maintenance, pollution, marital issues and many more...
- Migration is draining talent from rural or underdeveloped regions.
- Migrants can be made aware of the government schemes and can be helped to sustain themselves in the alien surroundings.

1.3 Fact-Finding Chart

Objective	Technique	Subjects	Time-commitment
To get the background Information	Background reading	Research papers ,articles	2 days
Introduce Government initiatives and collaborate them area-wise to help the migrating people	Interview	The government reports and surveys	project duration
Help collect information About problems faced by Migrants	Survey	The migrants	3-4 days
Age-migration trends	Background Reading, survey	The migrants, migrant records	1-2 days
Current information about the database	Interview	The director and the database administrator	30 minutes
To follow up on company information	Observation	The staff of the company	1 day
To establish the ideas To be added to the new system	Interview	2 managers and 3 creative staff	3 * 1 hours
To establish the Records and documents Already present with the company	Interview/ Document sampling	Filing clerk / Librarian	2 * 1 hours
Gender ratio in migration	Surveys, migration reports	Compare gender distribution in migrants.	1-2 days
Education level and Migration	Surveys, Background Reading	The migrants	1-2 days
Talent drain from rural/ underdeveloped regions	Assess the impact of migration	Regional development reports, surveys	1-2 days

1.4 Requirements List

List of the requirements gathered

1. Age Range of the migrated people
2. Gender of the migrated people
3. Education level of the migrated people
4. Occupation of the migrated people (optional)
5. Place of origin
6. Current location
7. Duration at migrated location (patterns)
8. Frequency of migration (patterns)
9. Reasons for migration
10. Challenges faced after migration

Requirements gathered from Observations

- Age-based migration patterns
- Gender ratio analysis
- Education level of the migrated people
- Distance of migration (mostly used to find if the migration is intrastate or interstate)
- Recent migration trends
- Frequency of migration
- Primary reasons of migration
- Challenges faced by the migrants
- Information on talent drain
- Sustaining migrants in new environments

1.5 User Privileges

User Classes and Privileges

1) Database Administrator (DBA):

Role: Manages the entire database system, ensures data integrity, security and backups.

Privileges: Full access to the database tables, views, and system functions. Can create, modify, delete any data and assign privileges to other users.

2) Data Entry Operator:

Role: Enters data from research and surveys about youngsters' migration patterns.

Privileges: Insert and Update permissions on tables related to youngsters, migration events, and surveys, but no limited access to sensitive information such as user login details.

3) Researcher:

Role: Conducts data analysis and runs queries to extract insights on migration trends.

Privileges: Read-only access to all tables and views. Can perform SELECT operations but cannot insert, update, or delete data. Might have access to specific aggregated views to maintain privacy.

4) Government Policy Researcher:

Role: Accesses reports and high-level data on migration trends to inform policy decisions.

Privileges: Read-only access to summarized or aggregated views that highlight trends (e.g., migration rates by region, employment, education, etc.). Restricted access to raw data to protect individual privacy.

5) Survey Analyst:

Role: Works with raw data from surveys and interviews, analyzing factors that influence migration.

Privileges: Access to detailed survey tables, the ability to run complex queries, and potentially the ability to modify analysis tables (but not core migration or youngster data).

6) Public User:

Role: Accesses public reports or summary data via a front-end web interface.

Privileges: Read-only access to highly aggregated or anonymized reports. No access to raw data.

User Class	Privileges
DBA	ALL PRIVILEGES on all tables, views, and procedures, including CREATE, DROP, ALTER, GRANT, REVOKE
Data Entry Operator	INSERT, UPDATE on Youngster , MigrationEvent tables
Researcher	SELECT on all tables and views, no INSERT/UPDATE/DELETE
Government Policy Researcher	SELECT on the GovernmentPolicy table and related tables
Survey Analyst	Access only to survey data
Public User	SELECT on public aggregated views, no access to core data tables

Operating Environment

Hardware Requirements:

Database Server: 4-core CPU, 16GB RAM, 512GB SSD

Software Requirements:

DBMS: PostgreSQL

OS: Windows / Ubuntu

External Interfaces: User interfaces for data entry, researchers and public reports, Google forms for collecting information, Different websites for researchers

Chapter 2: Database Design

2.1 Noun Analysis

1) Noun Verb analysis

Nouns	Verbs
Migration	Track
India	Migrate
Youngster	Move
Government	Affect
Village	Seek
Town	Adapt
Region	Influence
Gender	Suffer
Age	Determine
State	Enable
Health	Discriminate
Climate	Differentiate
Marriage	Participate
Employment	Live

Education	Enroll
Social Security	Settle
People	Identify
Culture	Understand
Population	Address
Reason	Meet
Skills	Inform
Alien	Went
Policymakers	Started
Pollution	Reasoned
Migrant	Going through
Work	Determined
Response	Work on
Factors	Happening
Stresses	Get
Job creation	Help
Infrastructure	Gather
Research papers	Get benefitted
Research	Reading
Scope	Implement
Schemes	Analyze

Example	Form
System	Fulfilled
Information	Stayed
Survey	Migrated
Location	Adapted
Duration	Faced
Frequency	DisplayedFollow
Men	Find
Women	Make
Survey	Observe
Challenges	Settle
Group	Get to know
Number	Observe
Distance	Conduct
Purpose	Arrive
Questionnaire	Create
NGO	Deploy
Services	Access
Patterns	Identify
Policies	Order
Ratio	Provide

Adaptation	Offer
International level portal	Sent
Studies	Concludes
Domain	
Live database	
Underdeveloped regions	
Observation	
Time	
Lack of security	
Purpose	
Exclusion issues	
Talent	
Talent drain	
Functional requirements	
Non-functional requirements	
Government initiatives	
Regional development	
Job creation	
Needs	
Infrastructure planning	
Process	

Locality	
Opportunities	
System	
Scalability	
Reliability	
Usability	
Political	
Conflict	
Data	
Maintenance	
Religion	
Region	
Scope	
Project	
People	
Survey	
Issue	
Design	
Migration	
Reason	
People	

Project	
Drain	

List of selected nouns and verbs for the ER schema

Nouns	Verbs
Youngster	lives
Migration_Event	migrates
Region	applies
Employment	works
Education	participates
Climate	enrolls
Government_Policy	from
Health	located
Opportunities	present

List of the rejected nouns with reason

Noun	Reject Reason
People, Men , Women, Town, Location, Village, Locality, Talent, Population, Pollution, India, Culture, Ratio, Data, Needs, Services	General
Process, Work, Response, System, Researchers, Policymakers, Problem domain, Project, International level portal, Questionnaire, Stakeholders, NGO, Functional and Non-Functional Requirements, Scalability, Reliability, Availability, Security, Usability, Performance	Irrelevant
Patterns, Movement, Time, Distance, Climate Change, Exclusion issues	Vague
Age, Gender, Occupation, Location, Frequency, Cultural Difference, Health Maintenance, Social Security, Salary, Talent drain	Attributes

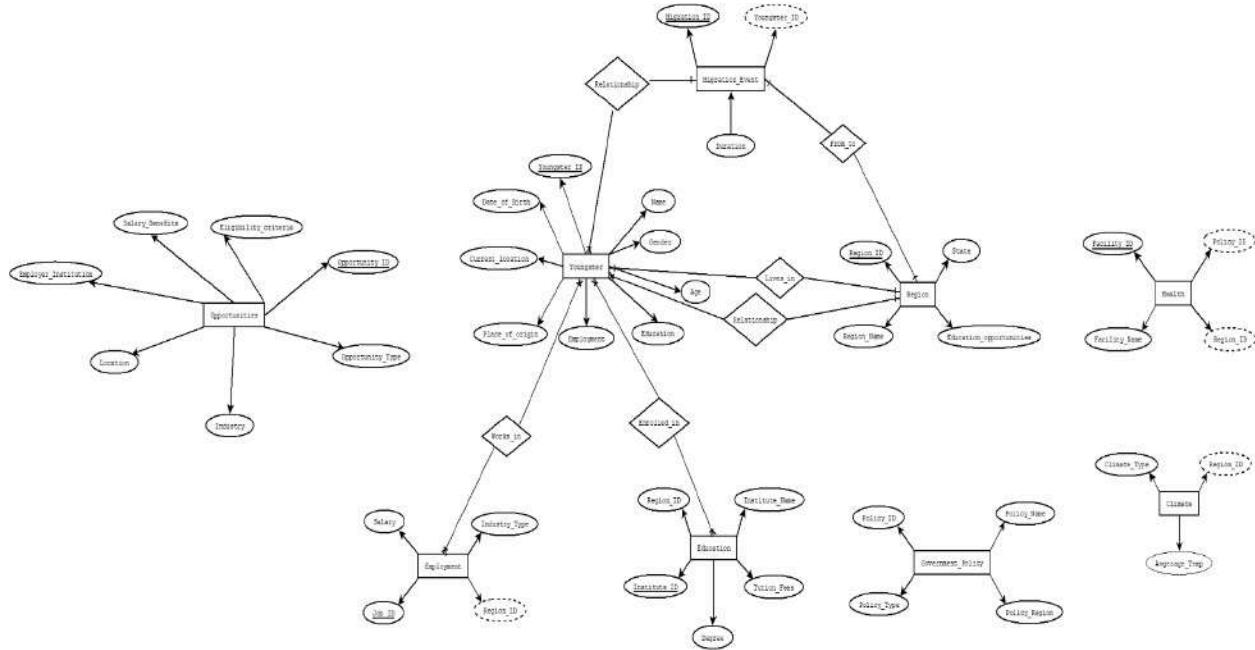
2.2 Schema and ER Diagram Design

Initial Schema

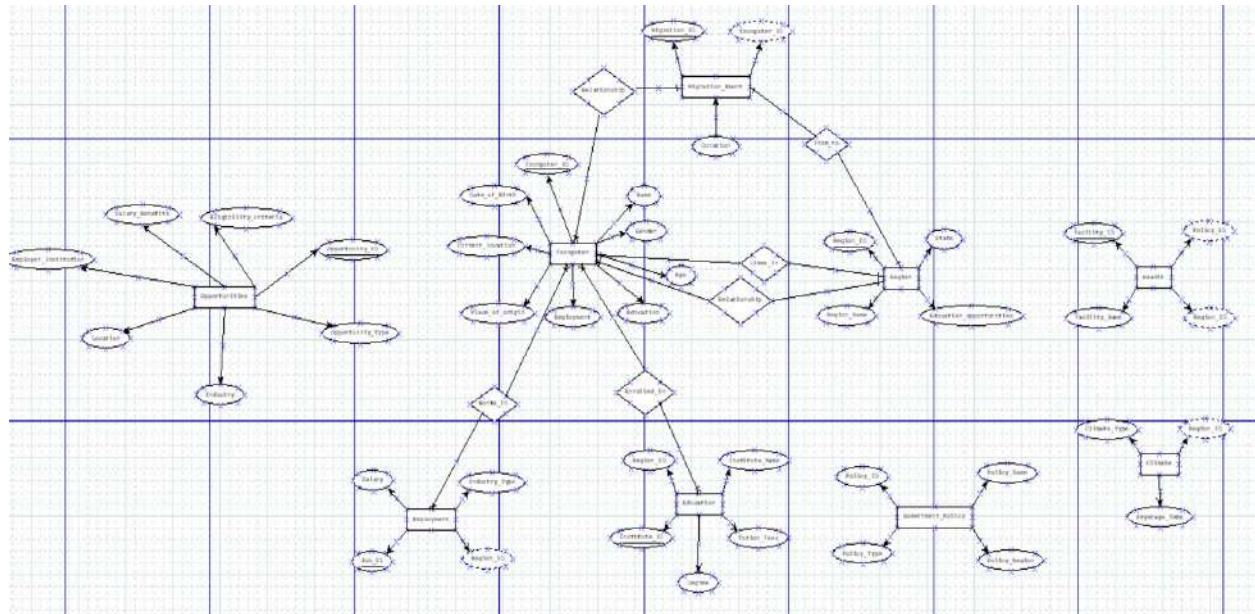
Candidate entity set	Candidate attribute set	Candidate relationship set
Youngster	<u>Youngster_ID</u> , name, gender, age, education, employment place_of_origin, Current_Location, Date_Of_Birth	Youngster lives_in Region (one to one), Youngster migrates_to Region (one to many), Youngster participates Migration_event (one to many), Youngster works_in Employment (many to many) Youngster enrolled_in Education (many to many)
Region	<u>Region_ID</u> , State, Region_Name, Education_Opportunities	Region has Climate
Migration_Event	<u>Migration_ID</u> Youngster_ID Duration	Migration_Event from_to Region,
Employment	<u>JOB_ID</u> <u>REGION_ID</u>	Employment related_to Region

	Industry_Type Salary	
Education	<u>Institute_ID</u> Region_ID Institute_Name Tution_Fees Degree	Education located_in Region
Climate	<u>Climate_Type</u> Region_ID Avgerage_Temp	Climate in Region
Government_Policy	<u>Policy_ID</u> Policy_Name Policy_Type Policy_Region	Government_Policy applies_to Region
Health	<u>Facility_ID</u> Policy_ID Facility_Name Region_ID	Health applies_to Government_Policy
Opportunities	<u>Opportunity_Type</u> , Industry Location Employer_Institution Salary_Benifits Eligibility_Criteria	Opportunities related_to Youngster , Opportunities present_in Region

Initial ER Diagram



Dia snapshot



2.3 ER Diagram Improvement

2.3.1 Identify Entity Types

1) Weak Entities:

Schema	Attributes
Climate	<u>Policy_ID</u> , <u>Region_ID</u> Avgerage_Temp, Climate_Type
Health	<u>Policy_ID</u> , <u>Region_ID</u> , Facility_Name

Climate can be modeled as a weak entity, dependent on Region, because climate conditions are specific.

Health can be modeled as another weak entity as health facilities and conditions may be region-specific.

Climate and Health are identified as weak entities and are combined together to depend on the region and are used to influence the government policies.

2) Type of relationships:

Hierarchy: Opportunities may involve a hierarchical relationship like it can contain different types of opportunities of job opportunities and educational opportunities. It can be subclassed into the two opportunities but it will complicate the diagram.

Aggregation: Climate and Health are weak entities that depend on Region. Since Government_Policy applies to both Climate and Health in a region, we have decided to model it using aggregation. It allows combining Climate and Health as dependent entities on a region associating them with government policies.

Recursive Relationship: We tried to find a recursive relationship in the schema but we do not have one.

Simple Association Link:

1. **Youngster lives_in Region:** 1:1 simple association
2. **Youngster migrates_to Region:** 1 simple association
3. **Youngster participates_in Migration_Event:** 1 simple association
4. **Youngster works_in Employment:** M simple association
5. **Youngster enrolled_in Education:** M simple association
6. **Region has Climate/Health:** 1 simple association
7. **Government_Policy applies_to Climate/Health:** M simple association
8. **Opportunities related_to Youngster:** M simple association
9. **Opportunities present_in Region:** M simple association

2.3.2 Identify relationship types:

1) Analysis of the diagram:

Entity vs Attribute:

We modeled Education and Employment as different entities rather than the attributes of a Youngster. We did this because Education and Employment involve many-to-many relationships with multiple entities, so it's suitable to make them entities instead of attributes.

Entity vs Relationship:

Youngster lives_in Region is modeled as a relationship and not an attribute, to capture a dynamic and potentially changing relationship.

Binary vs Ternary Relationships:

Binary relationships

Youngster lives_in Region - 1:1

Youngster migrates_to Region - 1:M

Youngster works_in Employment - M:M

Potential Ternary Relationship

Youngster migrates_to Region during Migration_Event

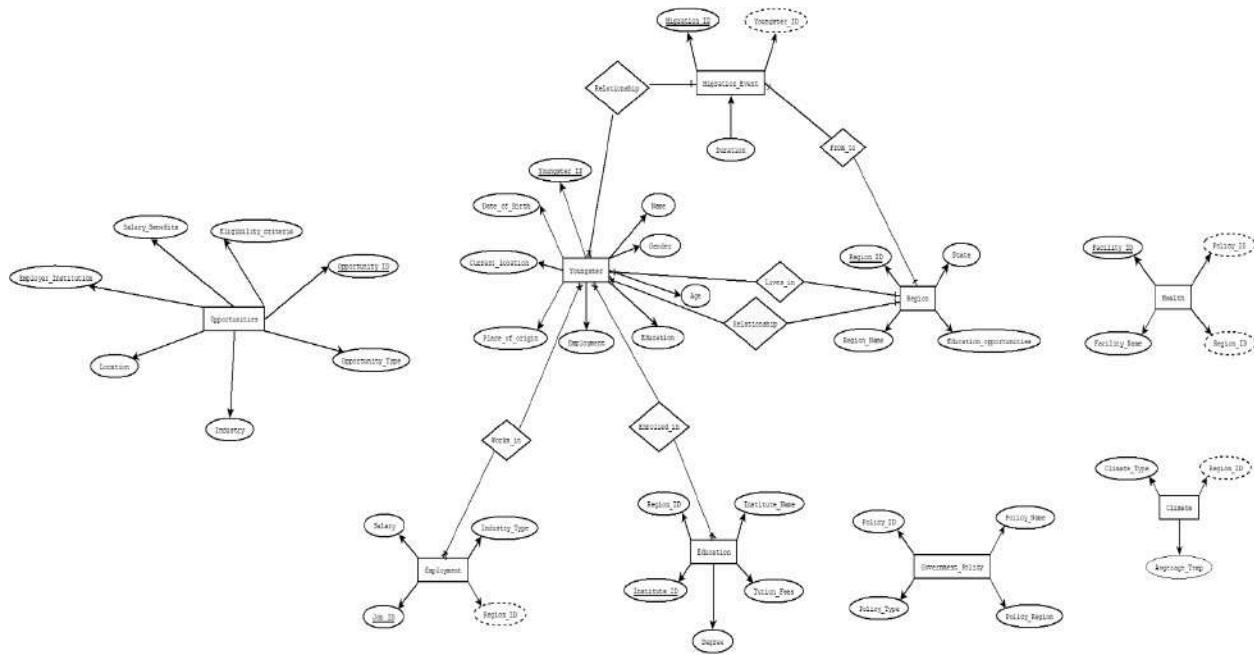
We avoid using this ternary relationship by using Youngster participates_in Migration_Event and Migration_event occurs_in Region

Aggregation vs Ternary Relationship:

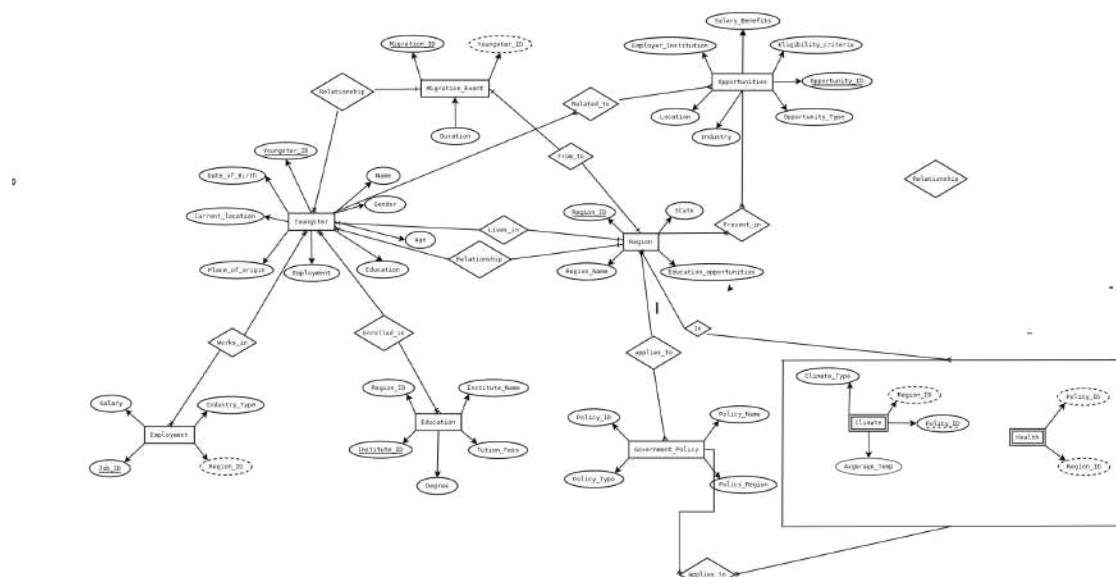
Aggregation of weak entities of Climate and Health is better than creating a ternary relationship

2.3.3 ER Diagram Analysis

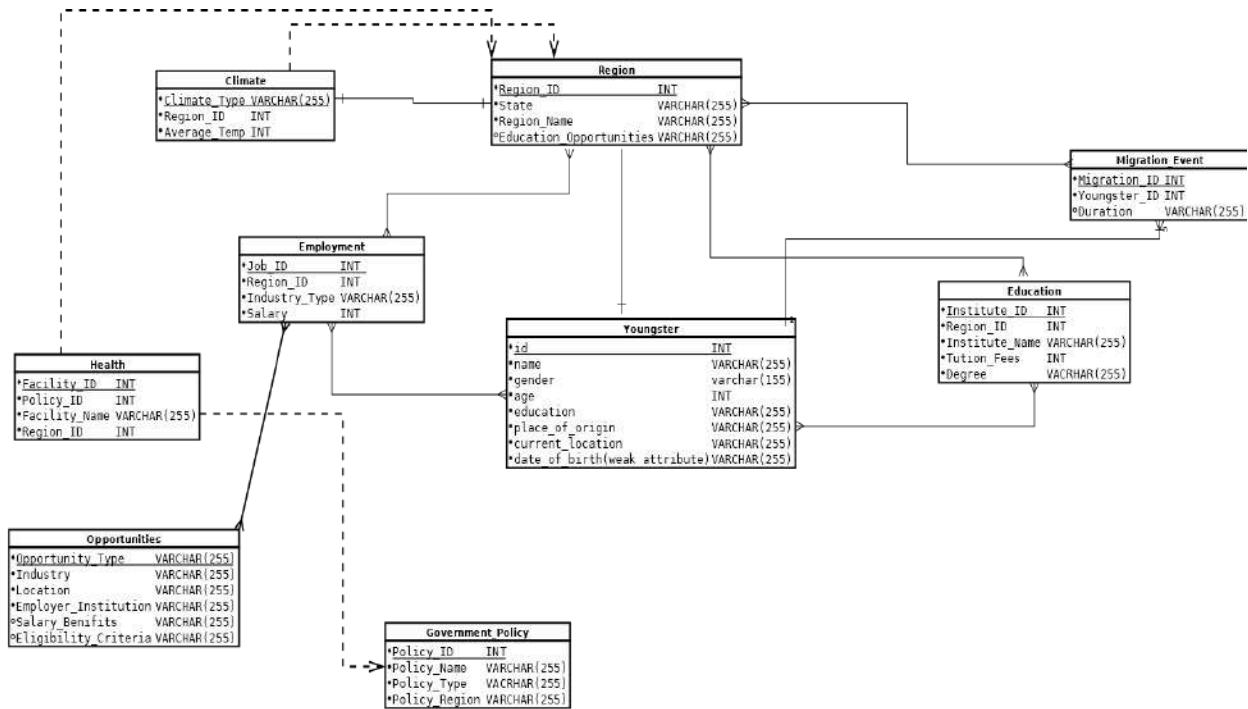
ER Version 1



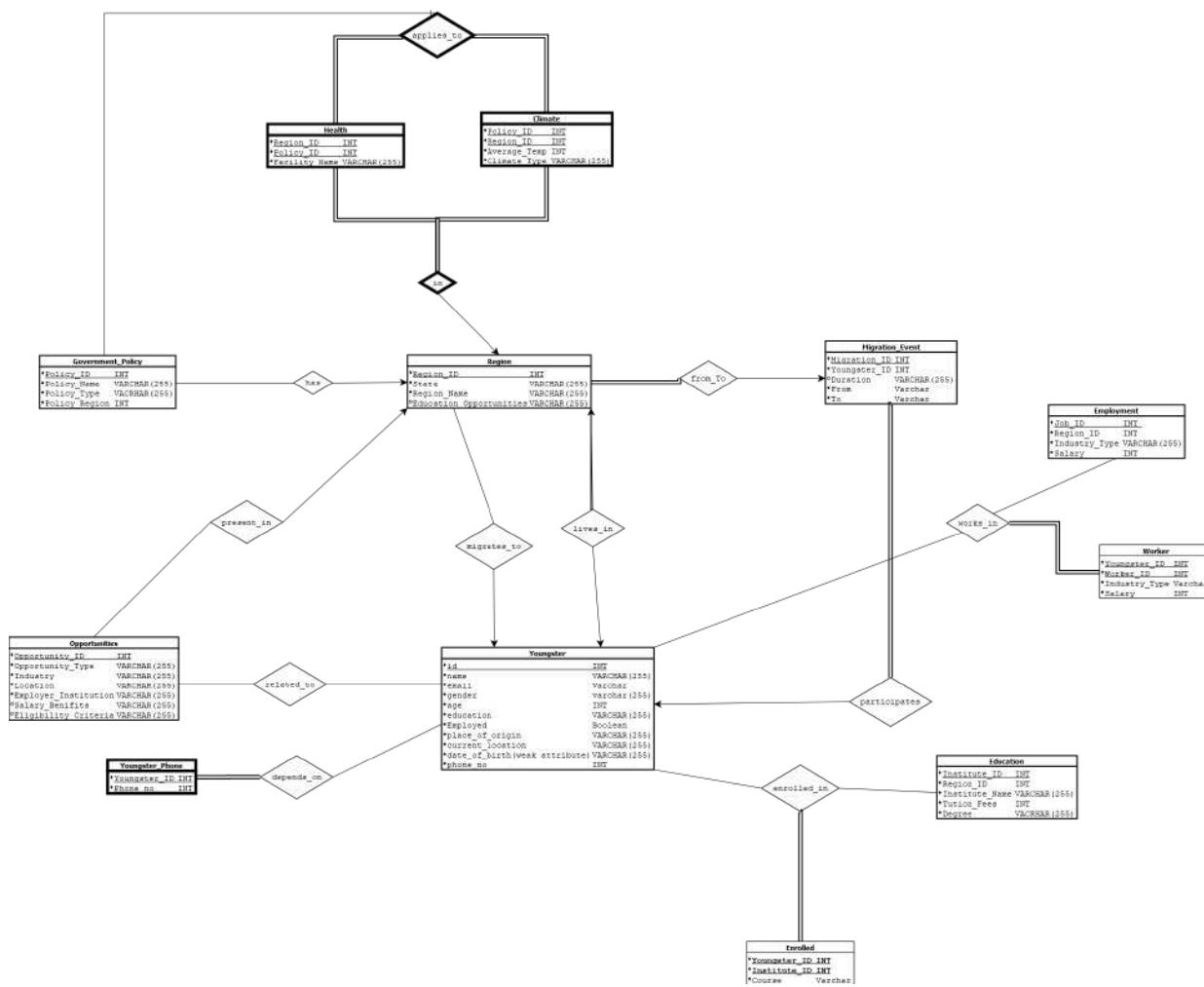
Updated ER Diagram (With aggregation changes) (ER Version 2)



Schema Diagram



2.4 Mapping ER Model to Relational Model



- Arrow and arrow refers to 1:1 relation
- Line and Line are N:M relation
- Arrow and line refers to 1:N relation
- Darker shades are weak objects
- Double lines represent total participation

Write down all the relations with the schema

- 1) Youngster (Youngster_ID, name, email, gender, age, education, employment place_of_origin, Current_Location, Date_Of_Birth, phone_no)
- 2) Region (Region_ID, State, Region_Type, Education_Opportunities)
- 3) Migration_Event (Migration_ID , Youngster_ID , Duration, From, To)
- 4) Employment (JOB_ID, Youngster_ID, Industry_Type, Salary)
- 5) Education (Institute_ID, Region_ID, Institute_Name Tution_Fees, Degree)
- 6) Climate (Policy_ID, Region_ID, Avgerage_Temp, Climate_Type)
- 7) Government_Policy (Policy_ID, Policy_Name, Policy_Type, Policy_Region)
- 8) Health (Region_ID, Policy_ID, Facility_Name)
- 9) Opportunities (Opportunity_Type, Industry, Location, Employer_Institution, Salary_Benifits, Eligibility_Criteria)
- 10) Worker (M to M relationship of Youngster and Employment) (Youngster_ID, Job_ID , Industry_Type, Salary)
- 11) Enrolled (M to M relationship of Youngster and Education) (Youngster_ID, Institute_ID, Course)
- 12) Youngster_Phone(Multivalued attribute)
(Youngster_ID, Phone_no)

2.5 Create DDL Scripts

i. Express following constraints (if applicable) in DDL:

2.5.1 Domain constraints

Youngster

Youngster_ID : Unique Identifier (positive integer)

Name: Varchar

Email: Varchar

Gender: Enum type ('Male', 'Female', 'Non-Binary')

Age: Integer

Education: Varchar

Employed: Boolean (Yes,No)

Place_of_Origin: Varchar

Current_Location: Varchar

Date_Of_Birth: Date

Phone_Number: INT

Region

Region_ID: Unique identifier

State: Varchar

Region_Type: Varchar ('Urban', 'Rural', 'Town')

Education_Opportunities: Varchar ('High', 'Medium', 'Low')

Migration_Event

Migration_ID: Unique identifier (positive integer)

Youngster_ID: Foreign key from Youngster(Youngster_ID)

Duration: Integer (Duration in months or years, should be positive)

From: Varchar

To: Varchar

Employment

Job_ID: Unique identifier (positive integer)

Youngster_ID: Foreign key from Youngster(Youngster_ID)

Industry_Type: Varchar

Salary: Integer

Education

Institute_ID: Unique identifier (Positive integer)

Region_ID: Foreign key from Region(Region_ID)

Institute_Name: Varchar

Tution_Fees: Integer

Degree: Varchar

Climate

Policy_ID: Foreign key from Government_Policy(Policy_ID)

Region_ID: Foreign key from Region(Region_ID)

Climate_Type: Varchar

Average_Temp: Integer

Health

Policy_ID: Foreign key from Government_Policy(Policy_ID)

Region_ID: Foreign key from Region(Region_ID)

Facility_Name: Varchar

Government_Policy

Policy_ID: Unique Identifier

Policy_Name: Varchar

Policy_Type: Varchar

Policy_Region: Foreign Key from Region(Region_ID)

Opportunities

Opportunity_ID: Unique Identifier

Opportunity_Type: Varchar

Industry: Varchar

Location: Varchar

Employer_Institution: Varchar

Salary_Benifits: Integer

Eligibility_Criteria: Varchar

Worker

Youngster_ID: Foreign key from Youngster (Youngster_ID)

Worker_ID: Foreign key from Employment (Job_ID)

Industry_Type: Varchar

Salary: INT

Enrolled

Youngster_ID: Foreign key from Youngster (Youngster_ID)

Institute_ID: Foreign key from Education (Institute_ID)

Course: Varchar

Youngster_Phone

Youngster_ID: Foreign key from Youngster (Youngster_ID)

Phone_no: INT

2.5.2 Key Constraints

Youngster

Primary key: Youngster_ID

No foreign keys

Region

Primary key: Region_ID

No foreign keys

Migration_Event

Primary key: Migration_ID

Foreign key: Youngster_ID references Youngster (Youngster_ID)

Employment

Primary key: Job_ID

Foreign key: Youngster_ID references Youngster(Youngster_ID)

Education

Primary key: Institute_ID

Foreign key: Youngster_ID references Youngster(Youngster_ID)

Climate

Primary key: (Policy_ID, Region_ID)

Foreign key: Policy_ID references Government_Policy (Policy_ID), Region_ID references Region(Region_ID)

Health

Primary key: (Policy_ID, Region_ID)

Foreign key: Policy_ID references Government_Policy (Policy_ID), Region_ID references Region(Region_ID)

Government_Policy

Primary key: Policy_ID

Foreign key: Policy_Region references Region(Region_ID)

Opportunities

Primary key: Opportunity_ID

Worker

Primary key: (Youngster_ID, Job_ID)

Foreign key: Youngster_ID references Youngster (Youngster_ID), Job_ID references Employment (Job_ID)

Enrolled

Primary key: (Youngster_ID, Institute_ID)

Foreign key: Youngster_ID references Youngster (Youngster_ID), Institute_ID references Education (Institute_ID)

Youngster_Phone

Primary key: (Youngster_ID, Phone_no)

Foreign key: Youngster_ID references Youngster (Youngster_ID)

Chapter 3: Normalization of Database

3.1 Normalization and Schema Refinement

3.1.1 Original Design of Database

List all relations and schemas with all details

- 1) Youngster (Youngster_ID, name, email, gender, age, education, employment_place_of_origin, Current_Location, Date_Of_Birth)
- 2) Region (Region_ID, State, Region_Type, Education_Opportunities)
- 3) Migration_Event (Migration_ID , Youngster_ID , Duration, From, To)
- 4) Employment (JOB_ID, Youngster_ID, Industry_Type, Salary)
- 5) Education (Institute_ID, Region_ID, Institute_Name Tution_Fees, Degree)
- 6) Climate (Policy_ID, Region_ID, Average_Temp, Climate_Type)
- 7) Government_Policy (Policy_ID, Policy_Name, Policy_Type, Policy_Region)
- 8) Health (Region_ID, Policy_ID, Facility_Name)

9) Opportunities (**Opportunity_ID**, Opportunity_Type, Industry, Location, Employer_Institution, Salary_Benifits, Eligibility_Criteria)

10) Worker (M to M relationship of Youngster and Employment)
(Youngster_ID, Job_ID)

11) Enrolled (M to M relationship of Youngster and Education)
(Youngster_ID, Institute_ID, Course)

12) Youngster_Phone(Multivalued attribute)
(Youngster_ID, Phone_no)

3.1.2 Dependency Analysis

Identify and list all types of dependencies (PK, FK, Functional Dependencies) for each relation

1) Youngster Table

PK: Youngster_ID

Attributes: name,email, gender, age, education, employment place_of_origin, Current_Location, Date_Of_Birth, phone_no

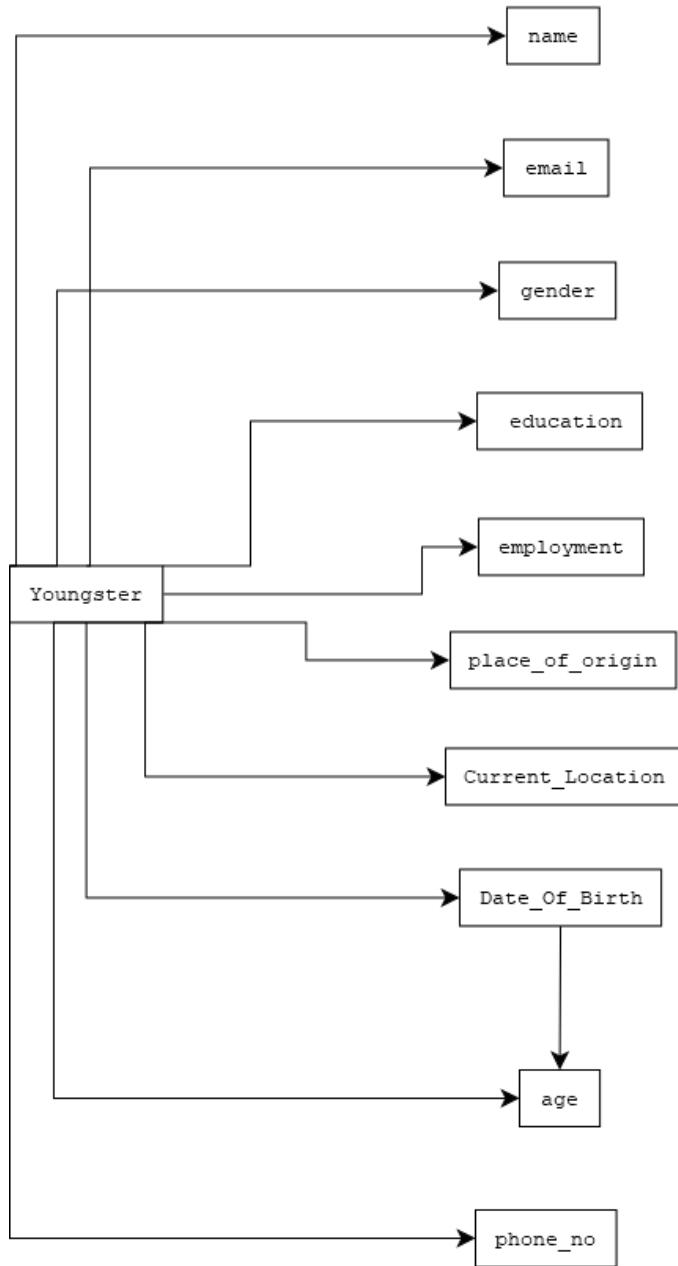
Functional dependencies:

Youngster_ID → name,email, gender, age, education, employment place_of_origin, Current_Location, Date_Of_Birth, phone_no

Derived Dependency:

Date_of_birth → Age

Reason: The attributes are dependent on the primary key and any changes in that will affect the other attributes of the table



Dealing with them: As we can calculate age using date of birth we can remove the age attribute and calculate age using SQL queries whenever needed

2) Region Table

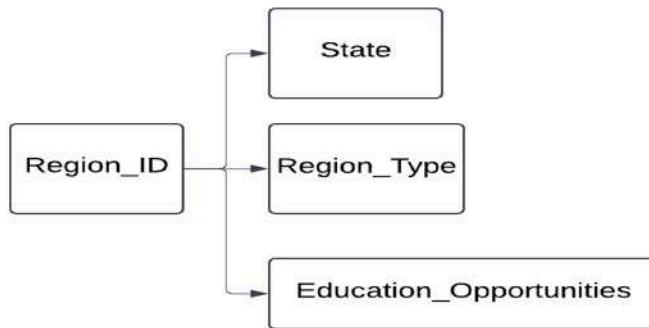
PK: Region_ID

Attributes: State, Region_Type, Education_Opportunities

Functional dependencies:

Region_ID → State, Region_Type, Education_Opportunities

Reason: The attributes are dependent on the primary key and any changes in that will affect the other attributes of the table



3) Migration_Event Table

PK: Migration_ID

FK: Youngster_ID

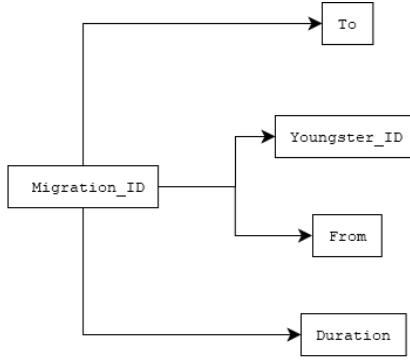
Attributes: Duration, From, To

Functional dependencies:

Migration_ID → Duration, From, To

Youngster_ID → Duration, From, To

Reason: The attributes are dependent on the PK,FK and any changes in that will affect the other attributes of the table



4) Employment

PK: `JOB_ID`

FK: `Youngster_ID`

Attributes: `Industry_Type`, `Salary`

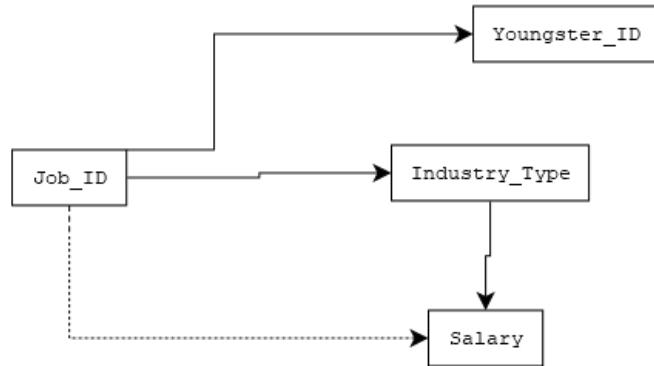
Functional dependencies:

`Job_ID` → `Industry_Type`

`Youngster_ID` → `Industry_Type`

`Industry_Type` → `Salary`

Reason: The attributes are dependent on the PK, FK and any changes in that will affect the other attributes of the table



Dealing with dependencies: As there is a transitive dependency of salary on `Job_ID` it creates confusion so we will store it in a Worker table table with all details about it there.

5) Education

PK: Institute_ID

FK: Region_ID

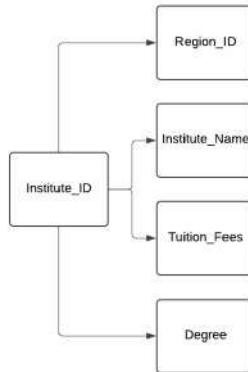
Attributes: Institute_Name, Tuition_Fees, Degree

Functional dependencies:

Institute_ID \rightarrow Institute_Name, Tuition_Fees, Degree

Region_ID \rightarrow Institute_Name, Tuition_Fees, Degree

Reason: The attributes are dependent on the PK, FK and any changes in that will affect the other attributes of the table



6) Climate

PK: (Policy_ID, Region_ID)

FK: Policy_ID, Region_ID

Attributes: Average_Temp, Climate_Type

Functional dependencies:

(Policy_ID, Region_ID) \rightarrow Average_Temp, Climate_Type

Reason: The attributes are dependent on the PK, FK and any changes in that will affect the other attributes of the table



7) Government_Policy

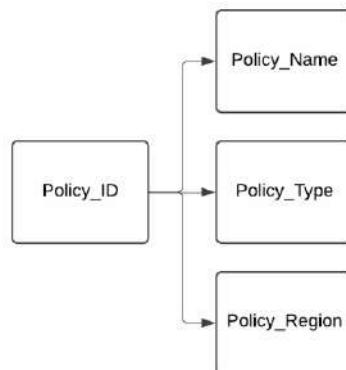
PK: Policy_ID

Attributes: Policy_Name, Policy_Type, Policy_Region

Functional dependencies:

Policy_ID —> Policy_Name, Policy_Type, Policy_Region

Reason: The attributes are dependent on the primary key and any changes in that will affect the other attributes of the table



8) Health

PK: (Region_ID, Policy_ID)

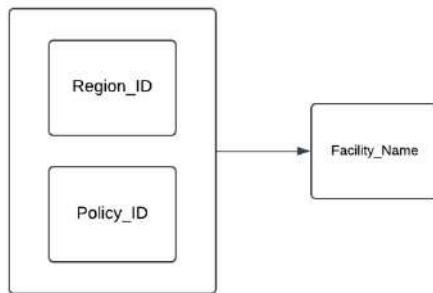
FK: Region_ID, Policy_ID

Attributes: Facility_Name

Functional dependencies:

(Region_ID, Policy_ID) \rightarrow Facility_Name

Reason: The attributes are dependent on the PK, FK and any changes in that will affect the other attributes of the table



9) Opportunities

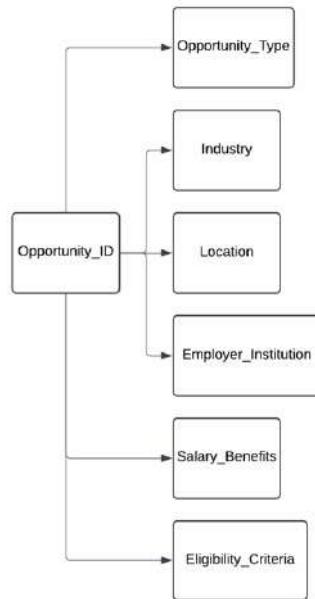
PK: Opportunity_ID

Attributes: Opportunity_Type, Industry, Location, Employer_Institution, Salary_Benifits, Eligibility_Criteria

Functional dependencies:

Opportunity_ID \rightarrow Opportunity_Type, Industry, Location, Employer_Institution, Salary_Benifits, Eligibility_Criteria

Reason: The attributes are dependent on the primary key and any changes in that will affect the other attributes of the table



10) Worker

PK: (Youngster_ID, Job_ID)

FK: Job_ID, Youngste_IDr

Attributes: Industry_Type, Salary

Functional dependencies:

Youngster_ID \rightarrow Industry_Type, Salary

Job_ID \rightarrow Industry_Type, Salary

Reason: The attributes are dependent on the PK, FK and any changes in that will affect the other attributes of the table



11) Enrolled

PK: (Youngster_ID, Institute_ID)

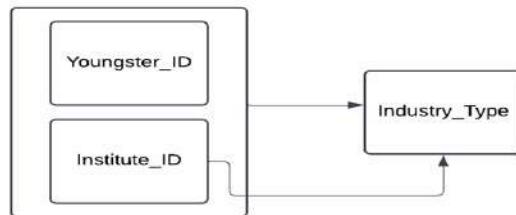
Attribute: Course

Functional dependencies:

Youngster_ID \rightarrow Course

Institute_ID \rightarrow Course

Reason: The attributes are dependent on the primary key and foreign key and any changes in that will affect the other attributes of the table

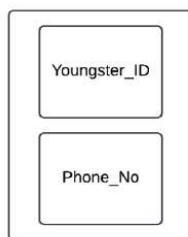


12) Youngster_Phone(Multivalued attribute)

PK: (Youngster_ID, Phone_no)

Functional dependencies: None

Reason: No other attributes



3.2 Redundancy and Anomalies Documentation

3.2.1 Redundancies

1) Youngster Table

- Current_Location and Place_Of-Origin might overlap with data in Region and Migration_Event

2) Employment Table

- Industry_Type and Salary are also listed in the Worker table, leading to duplication

3) Education Table

- Institute_Name and Tuition_Fees might be repeated if multiple youngsters enroll in the same institute.

4) Climate Table

- Region_ID is already present in the Region table.
- If Climate_Type or Average_Temp applies at the region level, it might be redundant to repeat it here.

5) Opportunities Table

- Location might overlap with Region_ID, and Employer_Institution might have repeated data if multiple opportunities are offered by the same employer.

6) Health Table

- Facility_Name could lead to redundancy if the same facility exists in multiple regions.

3.2.2 Anomalies

1) Youngster :

Update: If you change youngster's phone number you need to edit the whole row

Delete: Since Youngster's phone number is stored in another table updating the number you will have to update the whole row

2) Employment:

Update: There could be many opportunities with the same name so updating them would also update other attributes

Delete: There could be many opportunities with the same name so deleting them would also delete other attributes

3) Education:

Update: If we change tuition fees of a institute where multiple youngster's are enrolled we will have to make relevant changes in all the tables

Delete: Deleting an institute will result in its history with the youngster's being deleted

Insert: You can't insert a new record if you don't have all the details (tuition_fees, degree)

4) Health:

Update: Facilities having same name would require changes across multiple rows

Delete: Facilities having same name would require deletion across multiple rows

Insert: Facilities having same name would require insertion across multiple rows

5) Opportunities:

Update: Whenever we update employer institution it updates all other attributes

Delete: Whenever we delete employer institution it delete all other attributes

3.3 Normalization Process

3.3.1 1NF

Normalize the database up to 1NF (scalar values)

Youngster Table:

- Move Phone_no to a separate table (Youngster_Phone) to handle the multi-valued attribute and create a 1 relationship.
- Consider removing Current_Location and handling it through the Migration_Event table to track movement more efficiently.

Education Table:

- Normalize Institute_Name and Tuition_Fees by creating a separate Institute table (e.g., Institute(Institution_ID, Name, Tuition_Fees)), which can then be linked to the Region table.

Opportunities Table:

- Normalize Employer_Institution by creating a separate Employer table (e.g., Employer(Employer_ID, Name)), linking opportunities to this new entity.

Health Table:

- Normalize the Facility_Name into a separate Health_Facility table (e.g., Health_Facility(Facility_ID, Name)), which can be referenced by Region and Policy.

3.3.2 2NF

Problems encountered:

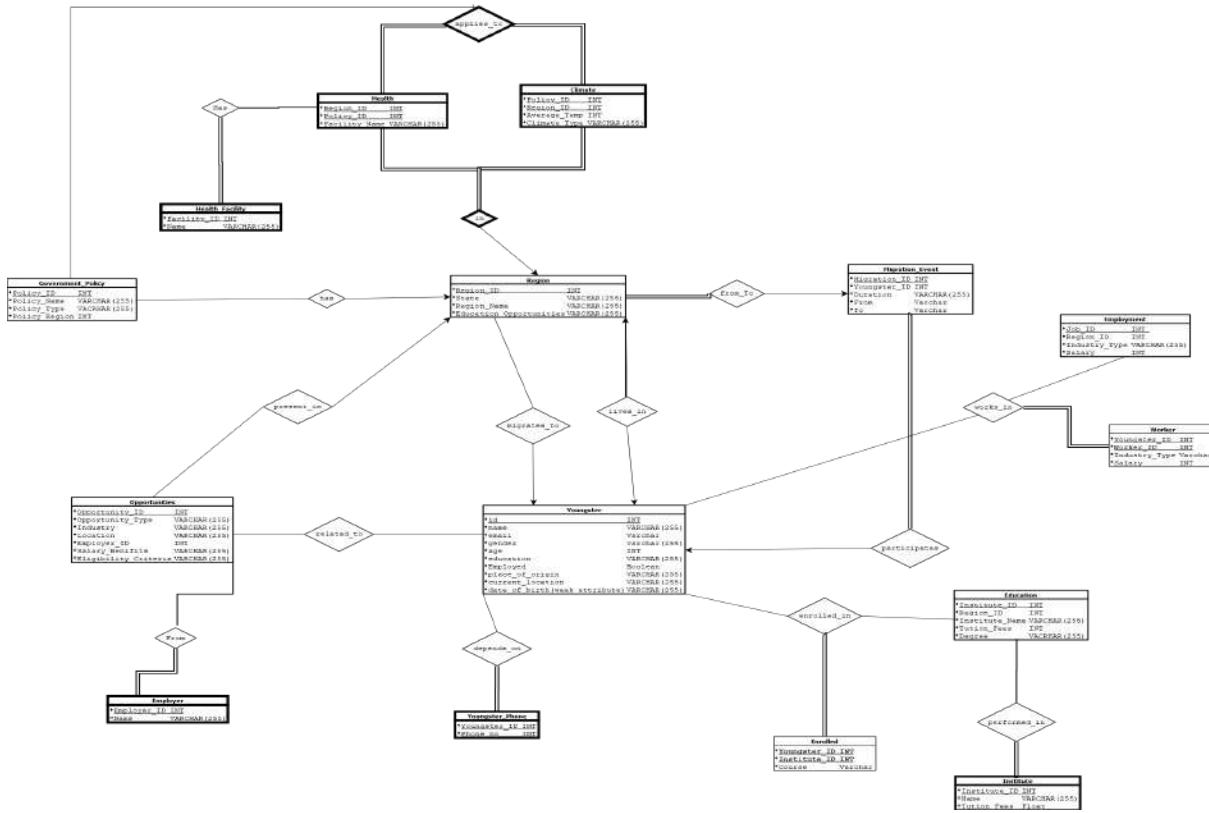
- 1) Multiple occurrence and irrelevance of Institute name and Tuition fees in the Education table.
- 2) Redundancy in the employer_Institution attribute of the Opportunities table.
- 3) Redundancy of Facility_Name in the Health table

However the tables are already in 2NF but we need to remove the redundancies that have occurred are to be removed.

Schema after normalization 2NF

- 1) Youngster (**Youngster_ID**, name, email, gender, education, employment place_of_origin, DateOfBirth,)
- 2) Region (**Region_ID**, State, Region_Type, Education_Opportunities)
- 3) Migration_Event (**Migration_ID** , Youngster_ID , Duration, From, To)
- 4) Employment (**JOB_ID**, Youngster_ID, Industry_Type, Salary)
- 5) Education (**Institute_ID**, Region_ID, Degree)
- 6) Climate (**Policy_ID**, **Region_ID**, Average_Temp, Climate_Type)

-
- 7) Government_Policy (**Policy_ID**, Policy_Name, Policy_Type, Policy_Region)
 - 8) Health (**Region_ID**, **Policy_ID**, **Facility_ID**)
 - 9) Opportunities (**Opportunity_ID**, Opportunity_Type, Industry, Location, Employer_ID, Salary_Benifits, Eligibility_Criteria)
 - 10) Worker (M to M relationship of Youngster and Employment)
(**Youngster_ID**, **Job_ID** , Industry_Type, Salary)
 - 11) Enrolled (M to M relationship of Youngster and Education)
(**Youngster_ID**, **Institute_ID**, Course)
 - 12) Youngster_Phone(Multivalued attribute) (**Youngster_ID**, Phone_no)
 - 13) Institute(**Institution_ID**, Name, Tuition_Fees)
 - 14) Employer(**Employer_ID**, Name)
 - 15) Health_Facility(**Facility_ID**, Name)



The changes made in 2NF were:

1. Removal of Institute name and tuition fees from Education table and creating a new table of Institute
2. Replacing the Employer_Institution in Opportunities table with Employer_ID and creating a separate table of Employer.
3. Creating a Health_Facility table and removing the facility name from the Health table.

3.3.3 3NF

Problems encountered:

- 1) Irrelevant and redundant Industry to make it into industry table

However the tables are already in 2NF but we need to remove the redundancies that have occurred are to be removed.

Schema after normalization 3NF

- 1) Youngster (**Youngster_ID**, name, email, gender, education, employment place_of_origin, DateOfBirth,)
- 2) Region (**Region_ID**, State, Region_Type, Education_Opportunities)
- 3) Migration_Event (**Migration_ID** , Youngster_ID , Duration, From, To)
- 4) Employment (**JOB_ID**, Youngster_ID, Industry_Type, Salary)
- 5) Education (**Institute_ID**, Region_ID, Degree)
- 6) Climate (**Policy_ID, Region_ID**, Average_Temp, Climate_Type)
- 7) Government_Policy (**Policy_ID**, Policy_Name, Policy_Type, Policy_Region)
- 8) Health (**Region_ID, Policy_ID, Facility_ID**)

9) Opportunities (**Opportunity_ID**, Opportunity_Type, Location, Employer_ID, Salary_Benifits, Eligibility_Criteria)

10) Worker (M to M relationship of Youngster and Employment)
(**Youngster_ID**, **Job_ID** , Industry_Type, Salary)

11) Enrolled (M to M relationship of Youngster and Education)
(**Youngster_ID**, **Institute_ID**, Course)

12) Youngster_Phone(Multivalued attribute) (**Youngster_ID**, Phone_no)

13) Institute(**Institution_ID**, Name, Tuition_Fees)

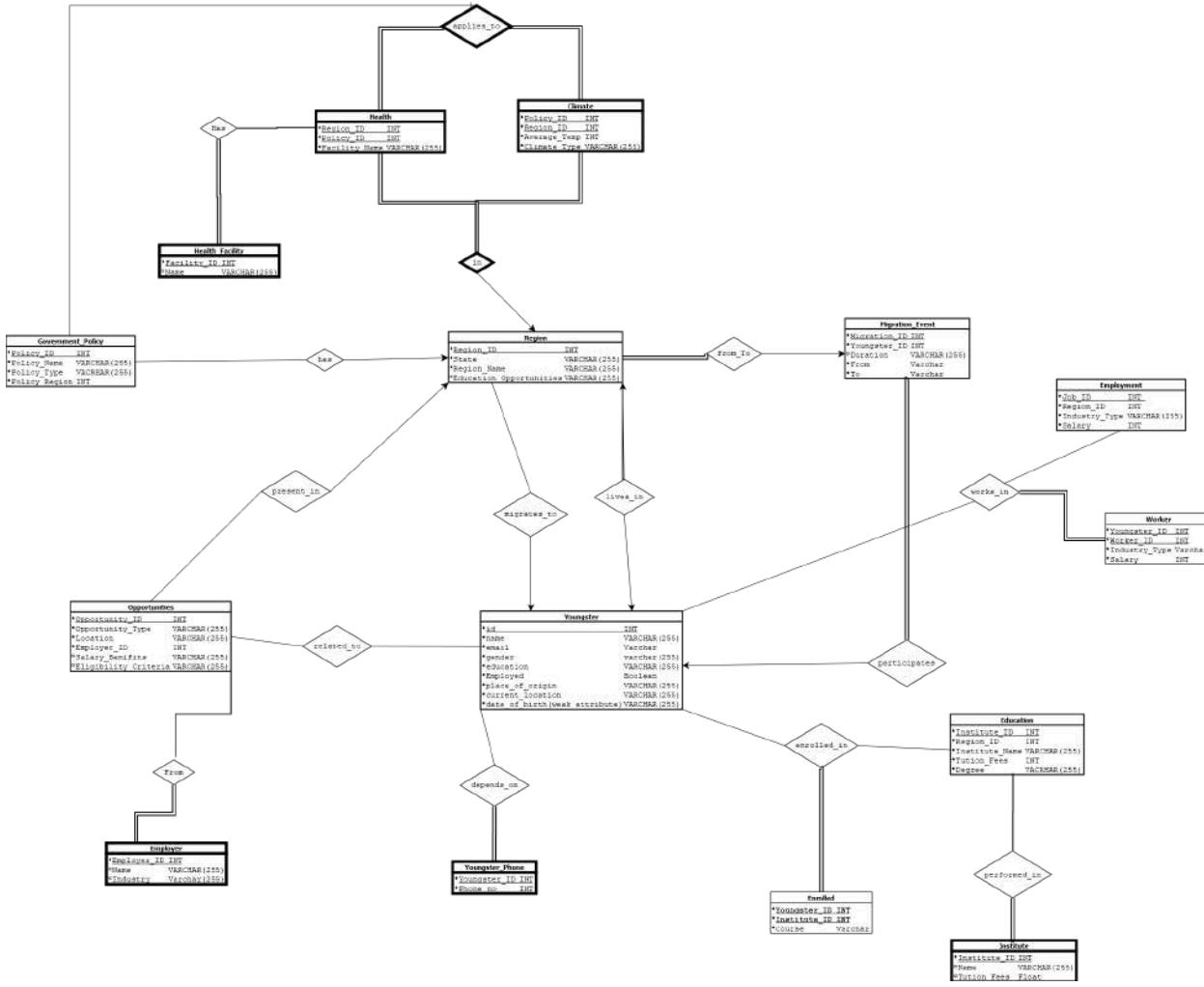
14) Employer(**Employer_ID**, Location, Salary_Benifits)

15) Health_Facility(**Facility_ID**, Name)

16) Industry (**Industry_Type**, Salary)

Changes made in 3NF were:

1. Normalized Industry into the Industry table, removing it from Opportunities.

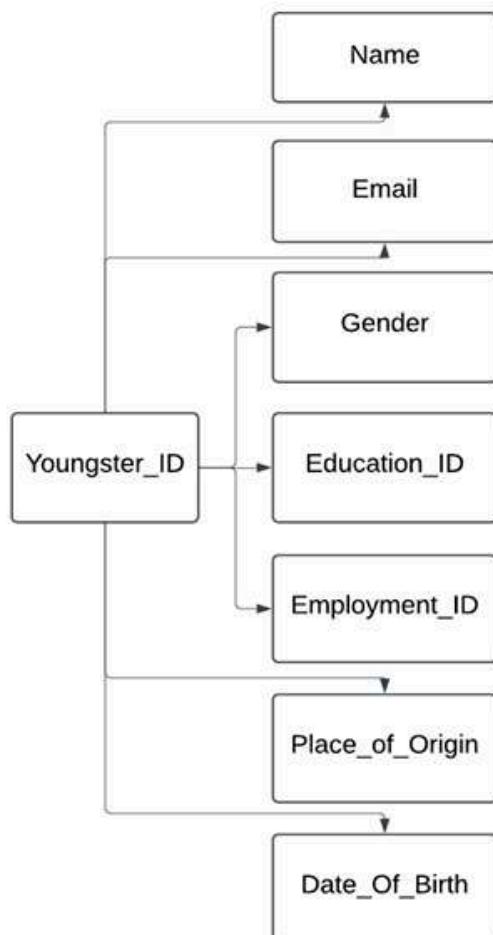


- After all the normalization we decided to make some changes to the schema to make it more understandable and better with adding some of the attributes and also checked everything for BCNF

Schema after normalization BCNF and added some attributes to tables that looked empty

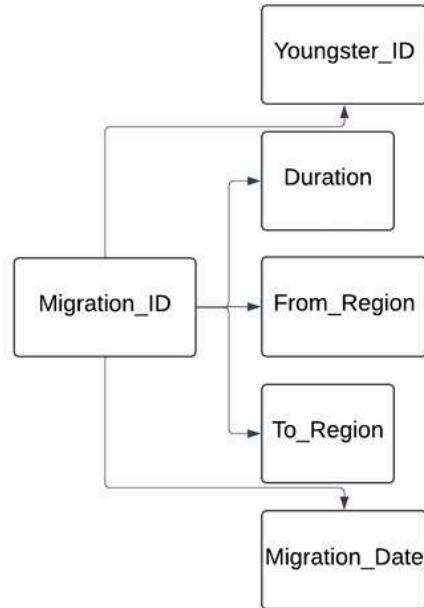
1) Youngster (Youngster_ID, name, email, gender, Education_ID, Employment_ID, place_of_origin, DateOfBirth)

1. $\text{Youngster_ID} \rightarrow \text{Name, Email, Gender, Education_ID, Employment_ID, Place_of_Origin, Date_Of_Birth}$



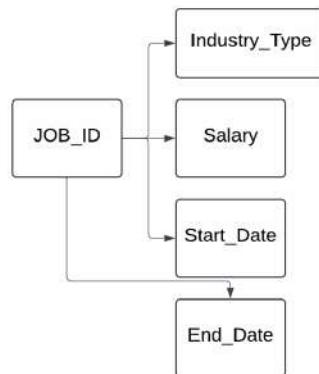
2) Migration_Event (**Migration_ID** , Youngster_ID , Duration,
From_Region, To_Region, Migration_Date)

3. $\text{Migration_ID} \rightarrow \text{Youngster_ID, Duration, From_Region, To_Region, Migration_Date}$

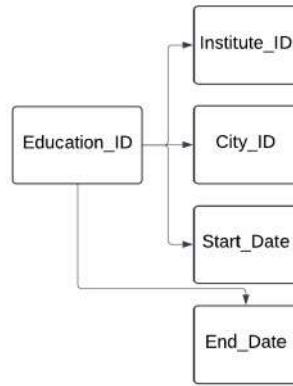


3) Employment (**JOB_ID**, Industry_Type, Salary, Start_Date,
End_Date)

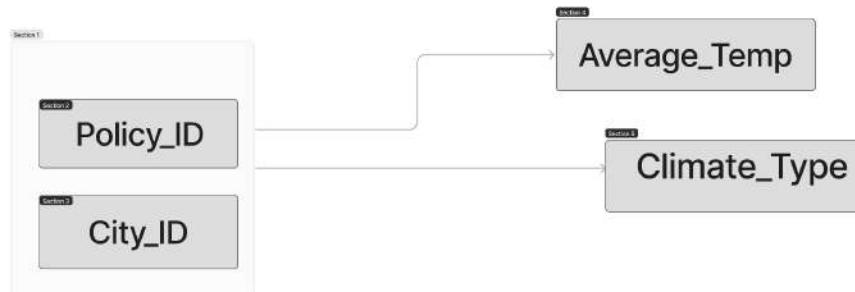
4. $\text{JOB_ID} \rightarrow \text{Industry_Type, Salary, Start_Date, End_Date}$



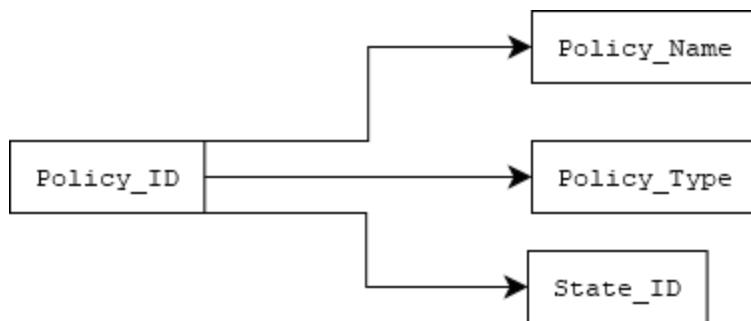
4) Education (**Education_ID** ,Institute_ID , City_ID, Start_Date, End_Date)



5) Climate (**City_ID**, **Policy_ID**, Average_Temp, Climate_Type)

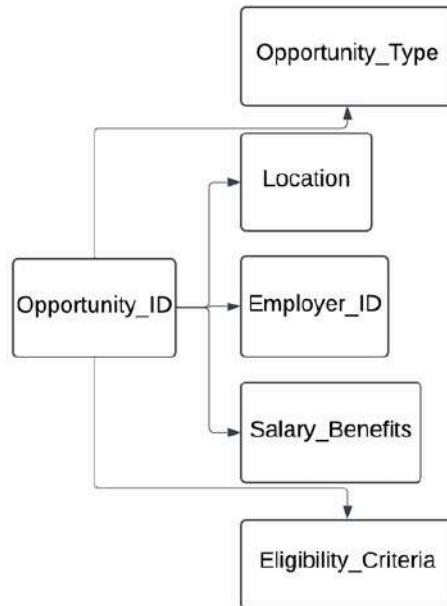


6) Government_Policy (**Policy_ID**, Policy_Name, Policy_Type, State_ID)



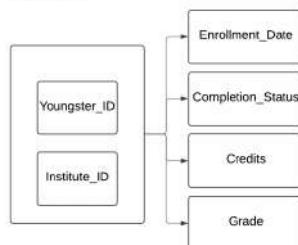
7) Opportunities (*Opportunity_ID*, Opportunity_Type, Location, Employer_ID, Salary_Benifits, Eligibility_Criteria)

6. $\text{Opportunity_ID} \rightarrow \text{Opportunity_Type}, \text{Location}, \text{Employer_ID}, \text{Salary_Benefits}, \text{Eligibility_Criteria}$

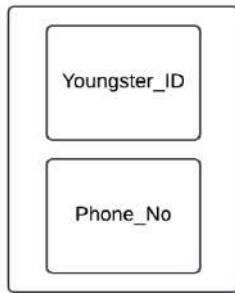


8) Enrolled (M to M relationship of Youngster and Education) Enrolled (*Youngster_ID*, *Institute_ID*, Enrollment_Date, Completion_Status, Credits, Grade)

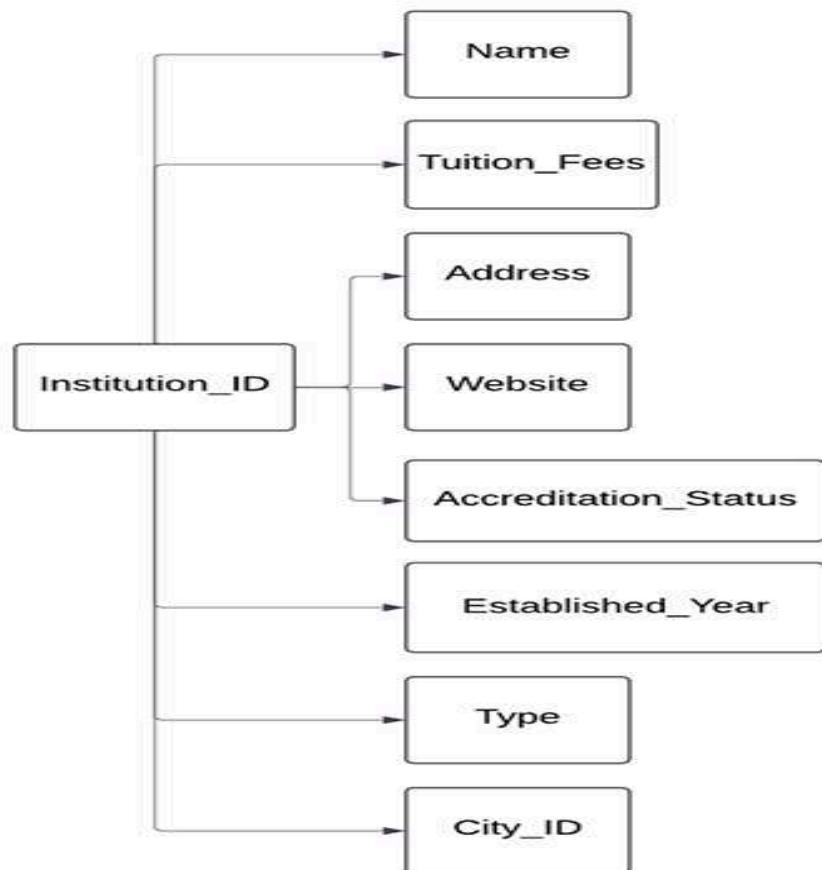
8. $\text{Youngster_ID}, \text{Institute_ID} \rightarrow \text{Enrollment_Date}, \text{Completion_Status}, \text{Credits}, \text{Grade}$



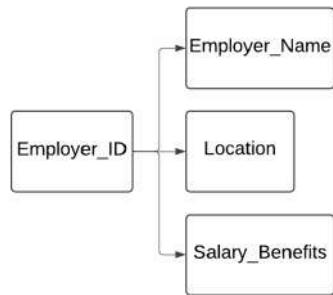
9) Youngster_Phone(Multivalued attribute) (**Youngster_ID**, Phone_no)



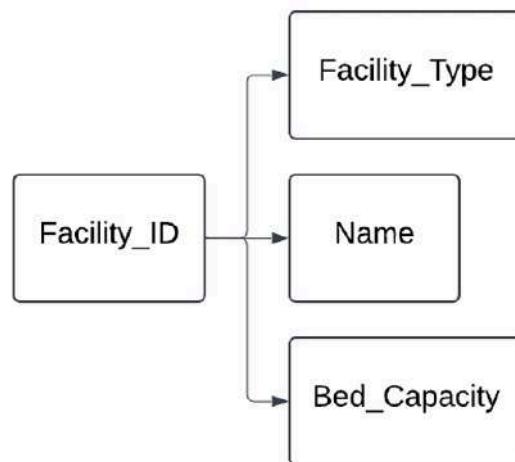
10) Institute(**Institution_ID**, Name, Tuition_Fees, Address, Website, Accreditation_Status, Established Year, Type, City_ID)



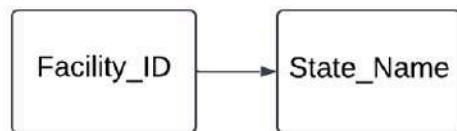
11) Employer(**Employer_ID**, Employer_Name, Location, Salary_Benefits)



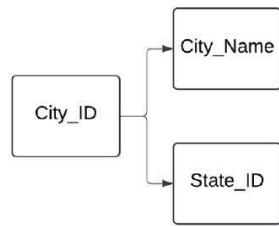
12) Health_Facility(**Facility_ID**, Name, Facility_Type, Bed_Capacity)



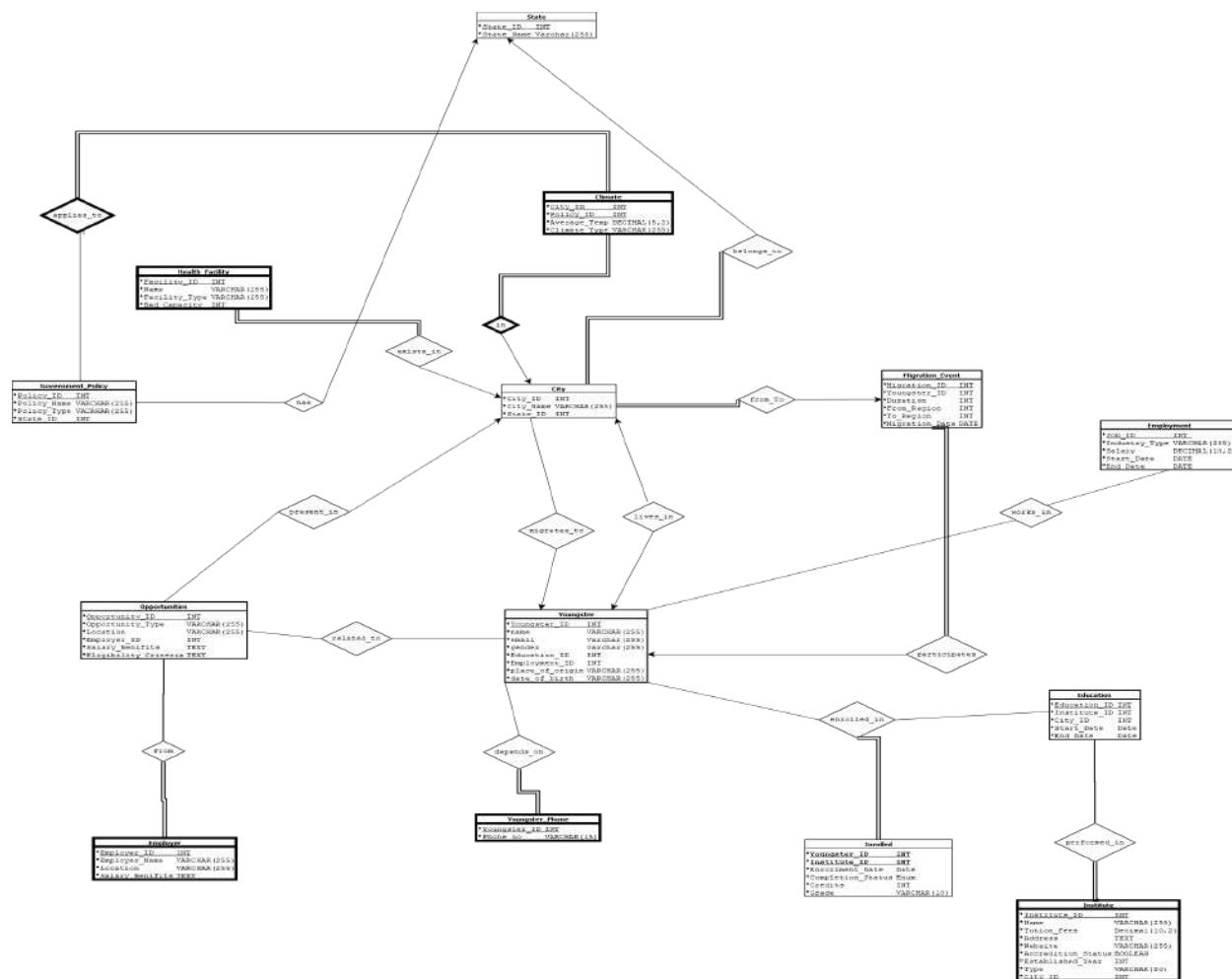
13) State (**State_ID**, State_Name)



14) City (**City_ID**, City_Name, State_ID)



Relational Diagram



Changes:

1. Added Education_ID and Employment_ID instead of education and employment for more clarity.
2. Removed Region table and added State and City table to be more clarified and all the relationships related to region were mapped to City table and the Government policy table relates to the State table.
3. Changed from and To to from_Region and To_Region for more clarification and added an attribute of migration_date to check the dates of migration.
4. Removed Youngster_ID from the table Employment as it was creating unnecessary redundancies and added start_date and end_date attributes to the table.
5. Created Education_ID for Education table and added all the important information of institute and degree to the Enrolled table and removed Region_ID foreign key, added City_ID as foreign key, and gave an Institute_ID foreign key to this table.
6. Removed Region_ID and added City_ID, City_ID and Policy_ID make a primary key keeping others the same.
7. Added State_ID to the government_Policy table to map it to the states in which they belong from the State table.

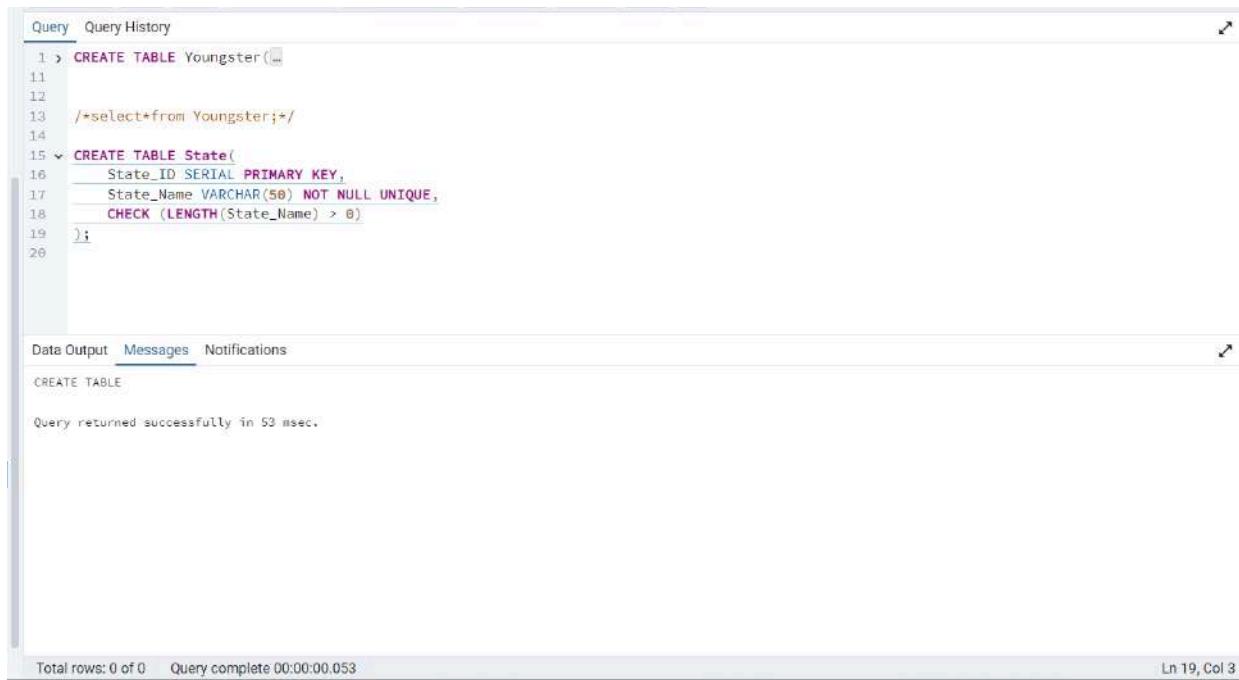
-
8. Removed the Health table as we already have the Health_Facility table as it was unnecessary and was creating redundancies.
 9. Removed the Worker table as it was unnecessary and was causing redundancies.
 10. Removed course attribute as it was causing redundancies and harming the entire schema and added all course detail attributes to the table of Enrollment_Date, Completion_Status, Credits, Grade.
 11. Added some more attributes to the institute table to make it more informative of Address, Website, Accreditation_Status, Established Year, Type City_ID is foreign key that maps the city of the institute.
 12. Added Employer_Name to the Employer_Table to make it informative.
 13. Added Facility_Type, Bed_Capacity attributes to the Health_Facility table to make it more informative.
 14. Removed the industry table as it was unnecessary and caused redundancies.

Chapter 4: Implementation of Database

4.1 Revised DDL Scripts

CREATE TABLE STATEMENTS

```
CREATE TABLE State (
    State_ID SERIAL PRIMARY KEY,
    State_Name VARCHAR(100) NOT NULL UNIQUE
);
```



The screenshot shows a database management interface with two main panes. The top pane is titled 'Query' and contains the SQL code for creating the 'State' table. The bottom pane is titled 'Messages' and displays the successful execution message.

```
Query  Query History
1 > CREATE TABLE Youngster(
11
12
13 /*select*from Youngster;*/
14
15 < CREATE TABLE State(
16     State_ID SERIAL PRIMARY KEY,
17     State_Name VARCHAR(50) NOT NULL UNIQUE,
18     CHECK (LENGTH(State_Name) > 0)
19 );
20

Data Output  Messages  Notifications
CREATE TABLE

Query returned successfully in 53 msec.

Total rows: 0 of 0  Query complete 00:00:00.053  Ln 19, Col 3
```

```

CREATE TABLE City (
    City_ID SERIAL PRIMARY KEY,
    City_Name VARCHAR(100) NOT NULL,
    State_ID INT NOT NULL,
    FOREIGN KEY (State_ID) REFERENCES State(State_ID) ON
    DELETE CASCADE
);

```



The screenshot shows a database query editor window. The 'Query' tab is active, displaying the SQL code for creating the 'City' table. The code includes constraints for the primary key, non-null city names, unique city names, non-null region IDs, non-null state IDs, and a check constraint for city names being longer than 6 characters. A foreign key constraint references the 'State' table's 'State_ID'. The 'Data Output' tab shows the successful execution of the query, indicating 0 rows affected and a completion time of 00:00:00.128. The status bar at the bottom right shows 'Ln 26, Col 51'.

```

Query  Query History
1 > CREATE TABLE Youngster();
11
12 > CREATE TABLE State();
17
18   select*from state;
19
20 < CREATE TABLE City(
21     City_ID SERIAL PRIMARY KEY,
22     City_Name VARCHAR(50) NOT NULL UNIQUE,
23     Region_ID INT NOT NULL,
24     State_ID INT NOT NULL,
25     CHECK(LENGTH(City_Name) > 6),
26     FOREIGN KEY (State_ID) REFERENCES State(State_ID)
27 );
28

Data Output  Messages  Notifications
CREATE TABLE

Query returned successfully in 128 msec.

Total rows: 0 of 0  Query complete 00:00:00.128
Ln 26, Col 51

```

```

CREATE TABLE Institute (
    Institution_ID SERIAL PRIMARY KEY,
    Name VARCHAR(255) NOT NULL,
    Tuition_Fees DECIMAL(10, 2) NOT NULL CHECK
    (Tuition_Fees >= 0),
    Address TEXT NOT NULL,
    Website VARCHAR(255),
    Accreditation_Status BOOLEAN NOT NULL,
    Established_Year INT CHECK (Established_Year > 0),
    Type VARCHAR(50),
    City_ID INT NOT NULL,

```

FOREIGN KEY (City_ID) REFERENCES State(City_ID) ON DELETE CASCADE);

```

Query  Query History
31 > CREATE TABLE Migration_Event(=
49
50 > CREATE TABLE Employment (=
60
61 < CREATE TABLE Institute (
62   Institution_ID SERIAL PRIMARY KEY,
63   Name VARCHAR(100) NOT NULL,
64   Tuition_Fees NUMERIC(10, 2) CHECK (Tuition_Fees >= 0),
65   Address VARCHAR(255) NOT NULL,
66   Website VARCHAR(100) UNIQUE CHECK (Website LIKE 'http%' AND LENGTH(Website) <= 100),
67   Accreditation_Status BOOLEAN NOT NULL,
68   Established_Year INT NOT NULL CHECK (Established_Year BETWEEN 1800 AND EXTRACT(YEAR FROM CURRENT_DATE)),
69   UNIQUE (Name, Address)
70 );
71

Data Output  Messages  Notifications
CREATE TABLE

Query returned successfully in 74 msec.

Total rows: 0 of 0  Query complete 00:00:00.074

```

Ln 68, Col 108

CREATE TABLE Employer (

Employer_ID SERIAL PRIMARY KEY,

Employer_Name VARCHAR(255) NOT NULL,

Location VARCHAR(255) NOT NULL,

Salary_Benefits TEXT NOT NULL

);

```

Query  Query History
75 > CREATE TABLE Education (=
87
88 > CREATE TABLE Climate (=
89
90 < CREATE TABLE Government_Policy (=
107
108 < CREATE TABLE Employer (
109   Employer_ID SERIAL PRIMARY KEY,
110   Location VARCHAR(100) NOT NULL,
111   Salary_Benefits NUMERIC(10, 2) CHECK (Salary_Benefits >= 0) NOT NULL,
112   Employer_Name VARCHAR(100) NOT NULL UNIQUE,
113   Established_Year INT CHECK (Established_Year BETWEEN 1900 AND EXTRACT(YEAR FROM CURRENT_DATE))
114
115 );
116

Data Output  Messages  Notifications
CREATE TABLE

Query returned successfully in 45 msec.

Total rows: 0 of 0  Query complete 00:00:00.045

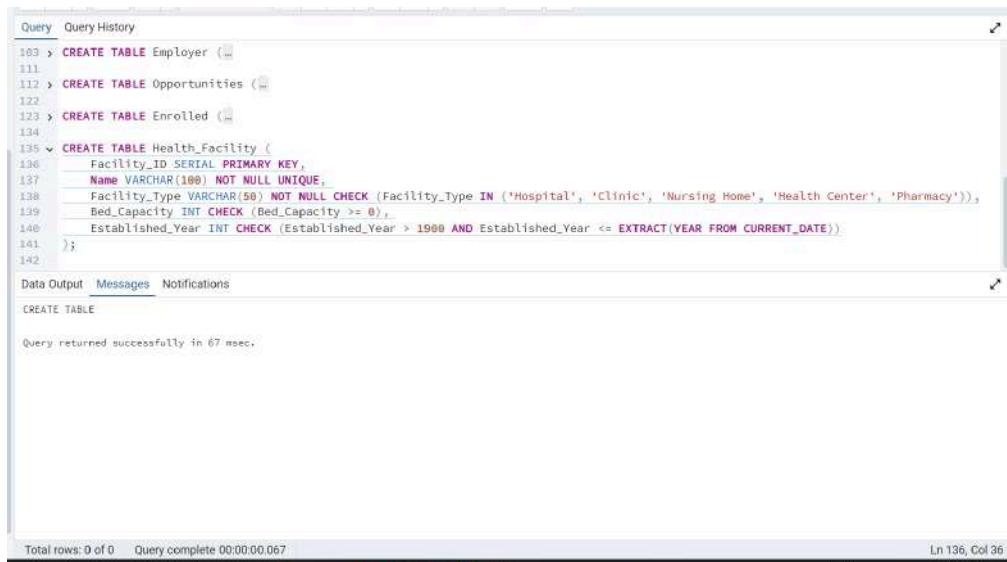
```

Ln 98, Col 1

```

CREATE TABLE Health_Facility (
    Facility_ID SERIAL PRIMARY KEY,
    Name VARCHAR(255) NOT NULL,
    Facility_Type VARCHAR(100) NOT NULL,
    Bed_Capacity INT NOT NULL CHECK (Bed_Capacity >= 0)
        Established_Year INT CHECK (Established_Year > 1900 AND
Established_Year <= Extract(YEAR FROM CURRENT_DATE))
);

```



The screenshot shows a database query editor window with the following details:

- Query History:** Shows several previous queries, including the creation of Employer, Opportunities, and Enrolled tables.
- Current Query:**

```

CREATE TABLE Health_Facility (
    Facility_ID SERIAL PRIMARY KEY,
    Name VARCHAR(100) NOT NULL UNIQUE,
    Facility_Type VARCHAR(50) NOT NULL CHECK (Facility_Type IN ('Hospital', 'Clinic', 'Nursing Home', 'Health Center', 'Pharmacy')),
    Bed_Capacity INT CHECK (Bed_Capacity >= 0),
    Established_Year INT CHECK (Established_Year > 1900 AND Established_Year <= EXTRACT(YEAR FROM CURRENT_DATE))
);

```
- Data Output:** Displays the message "CREATE TABLE" and "Query returned successfully in 67 msec."
- Bottom Status Bar:** Shows "Total rows: 0 of 0" and "Query complete 00:00:00.067".
- Bottom Right:** Shows "Ln 136, Col 36".

```

CREATE TABLE Government_Policy (
    Policy_ID SERIAL PRIMARY KEY,
    Policy_Name VARCHAR(255) NOT NULL,
    Policy_Type VARCHAR(100) NOT NULL,
    State_ID INT NOT NULL,
        FOREIGN KEY (State_ID) REFERENCES State(State_ID)
ON DELETE CASCADE
);

```

The screenshot shows a pgAdmin 4 interface with a query editor window. The title bar indicates the connection is to 'Youngsters_Migration_Database_db/postgres@PostgreSQL_16*'. The query editor contains several CREATE TABLE statements. The last statement shown is:

```
74 > CREATE TABLE Climate (
75   City_ID INT NOT NULL,
76   Policy_ID INT NOT NULL,
77   Average_Temp DECIMAL(5, 2) NOT NULL CHECK
78     (Average_Temp >= -50 AND Average_Temp <= 60),
79   Climate_Type VARCHAR(50) NOT NULL,
80   PRIMARY KEY (City_ID, Policy_ID),
81   FOREIGN KEY (City_ID) REFERENCES City(City_ID) ON
82   DELETE CASCADE
83 );
```

Below the query, the status bar shows 'Query returned successfully in 46 msec.' and 'Total rows: 0 of 0 Query complete 00:00:00.046'.

CREATE TABLE Climate (

City_ID INT NOT NULL,

Policy_ID INT NOT NULL,

Average_Temp DECIMAL(5, 2) NOT NULL CHECK

(Average_Temp >= -50 AND Average_Temp <= 60),

Climate_Type VARCHAR(50) NOT NULL,

PRIMARY KEY (City_ID, Policy_ID),

FOREIGN KEY (City_ID) REFERENCES City(City_ID) ON

DELETE CASCADE

);

The screenshot shows the pgAdmin 4 interface with a query editor window. The title bar indicates the connection is to 'Youngsters_Migration_Database_db/postgres@PostgreSQL 16+'. The query being run is:

```

67
68 CREATE TABLE Climate (
69     Climate_ID SERIAL PRIMARY KEY,
70     Region_ID INT NOT NULL,
71     Policy_ID INT,
72     Average_Temp NUMERIC(5, 2) NOT NULL CHECK (Average_Temp BETWEEN -50 AND 60),
73     Climate_Type VARCHAR(50) NOT NULL CHECK (Climate_Type IN ('Tropical', 'Arid', 'Semi-arid', 'Temperate', 'Mountain', 'Coastal')),
74     FOREIGN KEY (Region_ID) REFERENCES Region(Region_ID),
75     FOREIGN KEY (Policy_ID) REFERENCES Government_Policy(Policy_ID),
76     UNIQUE (Region_ID, Climate_Type)
77 );
78
79

```

The status bar at the bottom left shows 'Total rows: 0 of 0' and 'Query complete 00:00:00.080'. The status bar at the bottom right shows 'Ln 98, Col 1'.

CREATE TABLE Employment (

JOB_ID SERIAL PRIMARY KEY,

Industry_Type VARCHAR(100) NOT NULL,

Salary DECIMAL(10, 2) NOT NULL CHECK (Salary >= 0),

Start_Date DATE NOT NULL CHECK (Start_Date < CURRENT_DATE),

End_Date DATE CHECK (End_Date > Start_Date)

);

CREATE TABLE Education (

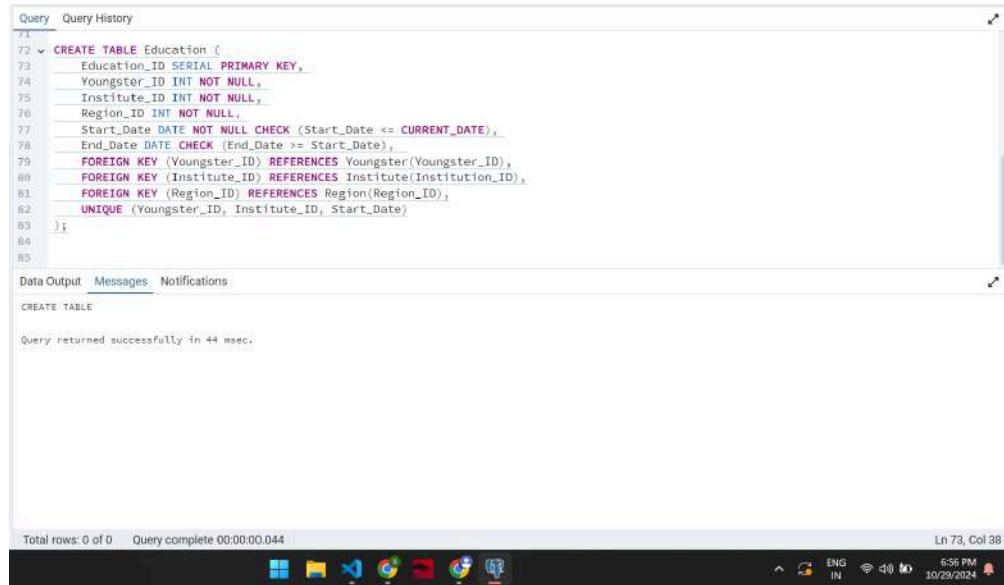
Education_ID SERIAL PRIMARY KEY,

Institute_ID INT NOT NULL,

City_ID INT NOT NULL,

Start_Date DATE NOT NULL CHECK (Start_Date < CURRENT_DATE),

End_Date DATE CHECK (End_Date > Start_Date),
 FOREIGN KEY (Institute_ID) REFERENCES
 Institute(Institution_ID) ON DELETE CASCADE,
 FOREIGN KEY (City_ID) REFERENCES City(City_ID) ON
 DELETE CASCADE
);



```

Query  Query History
71 CREATE TABLE Education (
72   Education_ID SERIAL PRIMARY KEY,
73   Youngster_ID INT NOT NULL,
74   Institute_ID INT NOT NULL,
75   Region_ID INT NOT NULL,
76   Start_Date DATE NOT NULL CHECK (Start_Date <= CURRENT_DATE),
77   End_Date DATE CHECK (End_Date >= Start_Date),
78   FOREIGN KEY (Youngster_ID) REFERENCES Youngster(Youngster_ID),
79   FOREIGN KEY (Institute_ID) REFERENCES Institute(Institution_ID),
80   FOREIGN KEY (Region_ID) REFERENCES Region(Region_ID),
81   UNIQUE (Youngster_ID, Institute_ID, Start_Date)
82 );
83
84
85

Data Output  Messages  Notifications
CREATE TABLE

Query returned successfully in 44 msec.

Total rows: 0 of 0  Query complete 00:00:00.044

```

The screenshot shows a SQL query window with the following details:

- Query History:** Shows the creation of the `Education` table.
- Data Output:** Shows the command `CREATE TABLE` and a success message indicating the query completed successfully in 44 msec.
- System Status:** Shows the system tray with icons for battery, signal, and date/time (10/29/2024, 6:56 PM).

CREATE TABLE Youngster (
 Youngster_ID SERIAL PRIMARY KEY,
 Name VARCHAR(100) NOT NULL,
 Email VARCHAR(100) NOT NULL UNIQUE,
 Gender VARCHAR(10) CHECK (Gender IN ('Male', 'Female',
 'Other')),
 Education_ID INT,
 Employment_ID INT,
 Place_of_Origin VARCHAR(255),
 Date_Of_Birth DATE NOT NULL CHECK (Date_Of_Birth <
 CURRENT_DATE)

FOREIGN KEY (Education_ID) REFERENCES
 Education(Education_ID) ON DELETE SET NULL,
 FOREIGN KEY (Employment_ID) REFERENCES
 Employment(JOB_ID) ON DELETE SET NULL
);



```

Query  Query History
1: CREATE TABLE Youngster(
2:   Youngster_ID SERIAL PRIMARY KEY,
3:   Name VARCHAR(50) NOT NULL,
4:   Email VARCHAR(100) NOT NULL UNIQUE,
5:   Gender CHAR(1) NOT NULL CHECK (Gender IN ('M','F','O')),
6:   Education_ID INT,
7:   Employment_ID INT,
8:   Place_of_Origin VARCHAR(50) NOT NULL,
9:   Date_of_Birth DATE NOT NULL CHECK (Date_of_Birth BETWEEN '1980-01-01' AND CURRENT_DATE)
10:
11:
12:

```

Data Output Messages Notifications

CREATE TABLE

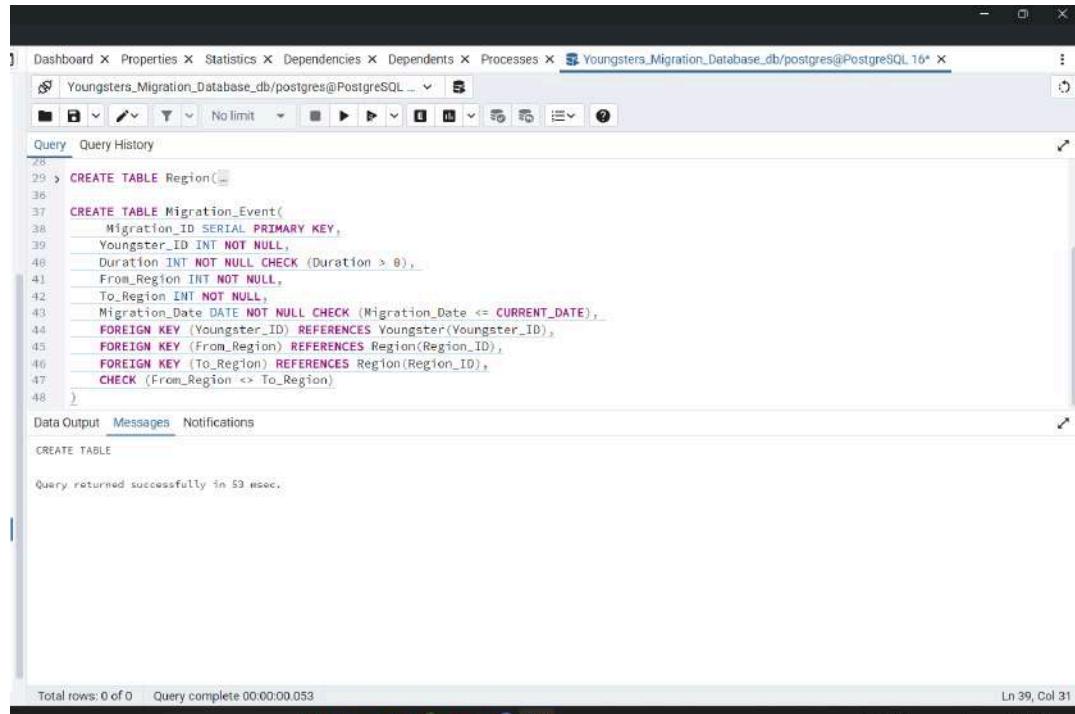
Query returned successfully in 40 sec.

Total rows: 0 of 0 Query complete 00:00:00.040 Ln 9, Col 20

CREATE TABLE Migration_Event (

Migration_ID SERIAL PRIMARY KEY,
 Youngster_ID INT NOT NULL,
 Duration INT NOT NULL CHECK (Duration >= 0),
 From_Region INT NOT NULL,
 To_Region INT NOT NULL,
 Migration_Date DATE NOT NULL CHECK (Migration_Date < CURRENT_DATE),
 FOREIGN KEY (Youngster_ID) REFERENCES
 Youngster(Youngster_ID) ON DELETE CASCADE,
 FOREIGN KEY (From_Region) REFERENCES
 City(City_ID) ON DELETE CASCADE,
 FOREIGN KEY (To_Region) REFERENCES City(City_ID)
 ON DELETE CASCADE

);



The screenshot shows a PostgreSQL query editor window. The title bar indicates the connection is to 'Youngsters_Migration_Database_db/postgres@PostgreSQL_16'. The main area displays the SQL code for creating the 'Migration_Event' table:

```
26 > CREATE TABLE Region();
27
28
29 > CREATE TABLE Migration_Event(
30     Migration_ID SERIAL PRIMARY KEY,
31     Youngster_ID INT NOT NULL,
32     Duration INT NOT NULL CHECK (Duration > 0),
33     From_Region INT NOT NULL,
34     To_Region INT NOT NULL,
35     Migration_Date DATE NOT NULL CHECK (Migration_Date <= CURRENT_DATE),
36     FOREIGN KEY (Youngster_ID) REFERENCES Youngster(Youngster_ID),
37     FOREIGN KEY (From_Region) REFERENCES Region(Region_ID),
38     FOREIGN KEY (To_Region) REFERENCES Region(Region_ID),
39     CHECK (From_Region <> To_Region)
40 );
41
42
43
44
45
46
47
48 )
```

Below the code, the status bar shows 'CREATE TABLE' and 'Query returned successfully in 53 msec.' The bottom status bar also shows 'Total rows: 0 of 0 Query complete 00:00:00.053' and 'Ln 39, Col 31'.

CREATE TABLE Opportunities (

```
    Opportunity_ID SERIAL PRIMARY KEY,  
    Opportunity_Type VARCHAR(100) NOT NULL,  
    Location VARCHAR(255) NOT NULL,  
    Employer_ID INT NOT NULL,  
    Salary_Benefits TEXT NOT NULL,  
    Eligibility_Criteria TEXT NOT NULL,  
    FOREIGN KEY (Employer_ID) REFERENCES  
    Employer(Employer_ID) ON DELETE CASCADE  
);
```

The screenshot shows the pgAdmin 4 interface with the 'Properties' tab selected. A query window is open with the following SQL code:

```

107 > CREATE TABLE Employer (
108     Employer_ID SERIAL PRIMARY KEY,
109     Opportunity_Type VARCHAR(50) NOT NULL CHECK (Opportunity_Type IN ('Job', 'Internship', 'Training', 'Contract', 'Volunteer')),
110     Location VARCHAR(100) NOT NULL,
111     Employer_ID INT NOT NULL,
112     Salary_Benefits NUMERIC(10, 2) CHECK (Salary_Benefits >= 0),
113     Eligibility_Criteria TEXT NOT NULL,
114     FOREIGN KEY (Employer_ID) REFERENCES Employer(Employer_ID),
115     UNIQUE (Opportunity_Type, Location)
116 );
117
118 < CREATE TABLE Opportunities (
119     Opportunity_ID SERIAL PRIMARY KEY,
120     Opportunity_Type VARCHAR(50) NOT NULL CHECK (Opportunity_Type IN ('Job', 'Internship', 'Training', 'Contract', 'Volunteer')),
121     Location VARCHAR(100) NOT NULL,
122     Employer_ID INT NOT NULL,
123     Salary_Benefits NUMERIC(10, 2) CHECK (Salary_Benefits >= 0),
124     Eligibility_Criteria TEXT NOT NULL,
125     FOREIGN KEY (Employer_ID) REFERENCES Employer(Employer_ID),
126     UNIQUE (Opportunity_Type, Location)
127 );
128

```

The status bar at the bottom indicates "Query returned successfully in 69 msec." and "Total rows: 0 of 0 Query complete 00:00:00.069".

CREATE TABLE Enrolled (

 Youngster_ID INT NOT NULL,

 Institute_ID INT NOT NULL,

 Enrollment_Date DATE NOT NULL CHECK

(Enrollment_Date < CURRENT_DATE),

 Completion_Status BOOLEAN NOT NULL,

 Credits INT NOT NULL CHECK (Credits >= 0),

 Grade VARCHAR(10),

 PRIMARY KEY (Youngster_ID, Institute_ID),

 FOREIGN KEY (Youngster_ID) REFERENCES

Youngster(Youngster_ID) ON DELETE CASCADE,

 FOREIGN KEY (Institute_ID) REFERENCES

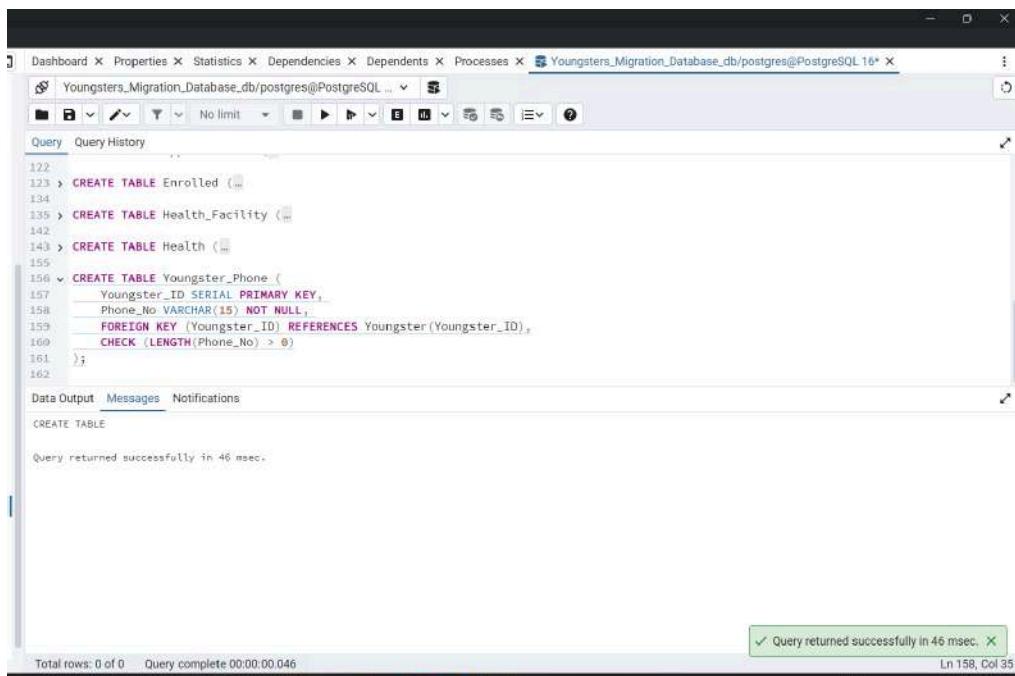
Institute(Institution_ID) ON DELETE CASCADE

);

```

CREATE TABLE Youngster_Phone (
    Youngster_ID INT NOT NULL,
    Phone_no VARCHAR(15) NOT NULL,
    PRIMARY KEY (Youngster_ID, Phone_No),
    FOREIGN KEY (Youngster_ID) REFERENCES
    Youngster(Youngster_ID) ON DELETE CASCADE
);

```



The screenshot shows the pgAdmin 4 interface with a query editor window. The query editor displays the SQL code for creating the `Youngster_Phone` table. The code includes constraints for the primary key (`Youngster_ID`), a foreign key reference to the `Youngster` table, and a check constraint ensuring the phone number length is greater than 0. A message at the bottom of the window indicates the query was executed successfully.

```

122
123 > CREATE TABLE Enrolled ...
134
135 > CREATE TABLE Health_Facility ...
142
143 > CREATE TABLE Health ...
155
156 < CREATE TABLE Youngster_Phone ...
157   Youngster_ID SERIAL PRIMARY KEY,
158   Phone_no VARCHAR(15) NOT NULL,
159   FOREIGN KEY (Youngster_ID) REFERENCES Youngster(Youngster_ID),
160   CHECK (LENGTH(Phone_no) > 0)
161 );
162
163
Data Output Messages Notifications
CREATE TABLE
Query returned successfully in 46 msec.

Total rows: 0 of 0 Query complete 00:00:00.046
✓ Query returned successfully in 46 msec. X
Ln 158, Col 35

```

4.2 Database Population

Insert Statements

1) State Table

```
INSERT INTO State (State_ID, State_Name) VALUES  
(1, 'Andhra Pradesh'),  
(2, 'Arunachal Pradesh'),  
(3, 'Assam'),  
(4, 'Bihar'),  
(5, 'Chhattisgarh'),  
(6, 'Goa'),  
(7, 'Gujarat'),  
(8, 'Haryana'),  
(9, 'Himachal Pradesh'),  
(10, 'Jharkhand'),  
(11, 'Karnataka'),  
(12, 'Kerala'),  
(13, 'Madhya Pradesh'),  
(14, 'Maharashtra'),  
(15, 'Manipur'),  
(16, 'Meghalaya'),  
(17, 'Mizoram'),  
(18, 'Nagaland'),
```

(19, 'Odisha'),
(20, 'Punjab'),
(21, 'Rajasthan'),
(22, 'Sikkim'),
(23, 'Tamil Nadu'),
(24, 'Telangana'),
(25, 'Tripura'),
(26, 'Uttar Pradesh'),
(27, 'Uttarakhand'),
(28, 'West Bengal'),
(29, 'Andaman and Nicobar Islands'),
(30, 'Chandigarh'),
(31, 'Dadra and Nagar Haveli and Daman and Diu'),
(32, 'Lakshadweep'),
(33, 'Delhi'),
(34, 'Puducherry'),
(35, 'Jammu and Kashmir'),
(36, 'Ladakh');

2) City Table

```
INSERT INTO City (City_ID, City_Name, State_ID) VALUES  
(1, 'Visakhapatnam', 1),  
(2, 'Vijayawada', 1),  
(3, 'Itanagar', 2),  
(4, 'Guwahati', 3),  
(5, 'Patna', 4),  
(6, 'Raipur', 5),  
(7, 'Panaji', 6),
```

-
- (8, 'Ahmedabad', 7),
(9, 'Gurugram', 8),
(10, 'Shimla', 9),
(11, 'Ranchi', 10),
(12, 'Bengaluru', 11),
(13, 'Thiruvananthapuram', 12),
(14, 'Bhopal', 13),
(15, 'Mumbai', 14),
(16, 'Imphal', 15),
(17, 'Shillong', 16),
(18, 'Aizawl', 17),
(19, 'Kohima', 18),
(20, 'Bhubaneswar', 19),
(21, 'Chandigarh', 20),
(22, 'Jaipur', 21),
(23, 'Gangtok', 22),
(24, 'Chennai', 23),
(25, 'Hyderabad', 24),
(26, 'Agartala', 25),
(27, 'Lucknow', 26),
(28, 'Dehradun', 27),
(29, 'Kolkata', 28),
(30, 'Port Blair', 29),
(31, 'Araku Valley', 1),
(32, 'Diu', 31),
(33, 'Kavaratti', 32),
(34, 'Delhi', 33),
(35, 'Puducherry', 34),
(36, 'Srinagar', 35),
-

-
- (37, 'Leh', 36),
(38, 'Jodhpur', 21),
(39, 'Ahmednagar', 14),
(40, 'Nashik', 14),
(41, 'Nagpur', 14),
(42, 'Dharmanagar', 25),
(43, 'Warangal', 24),
(44, 'Gwalior', 13),
(45, 'Kota', 21),
(46, 'Bikaner', 21),
(47, 'Jaisalmer', 21),
(48, 'Mysuru', 11),
(49, 'Belgaum', 11),
(50, 'Kochi', 12),
(51, 'Thrissur', 12),
(52, 'Mirzapur', 26),
(53, 'Dimapur', 18),
(54, 'Dibrugarh', 3),
(55, 'Bhatinda', 20),
(56, 'Ludhiana', 20),
(57, 'Vapi', 7),
(58, 'Vadodara', 7),
(59, 'Navi Mumbai', 14),
(60, 'Kalyan', 14),
(61, 'Cuttack', 19),
(62, 'Mangalore', 11),
(63, 'Rourkela', 19),
(64, 'Jamshedpur', 10),
(65, 'Dhanbad', 10),
-

-
- (66, 'Pithoragarh', 27),
(67, 'Kullu', 9),
(68, 'Siliguri', 28),
(69, 'Agra', 26),
(70, 'Varanasi', 26),
(71, 'Noida', 33),
(72, 'Ghaziabad', 26),
(73, 'Faridabad', 8),
(74, 'Baramulla', 35),
(75, 'Sagar', 13),
(76, 'Khandwa', 13),
(77, 'Sangli', 14),
(78, 'Kolhapur', 14),
(79, 'Aurangabad', 14),
(80, 'Nanded', 14),
(81, 'Jalna', 14),
(82, 'Kalyan-Dombivli', 14),
(83, 'Dharamshala', 9),
(84, 'Solan', 9),
(85, 'Palakkad', 12),
(86, 'Kottayam', 12),
(87, 'Tirupati', 1),
(88, 'Nellore', 1),
(89, 'Eluru', 1),
(90, 'Rangpo', 22),
(91, 'Dharamkot', 9),
(92, 'Bhuj', 7),
(93, 'Porbandar', 7),
(94, 'Udaipur', 21),
-

-
- (95, 'Ajmer', 21),
(96, 'Durgapur', 28),
(97, 'Bardhaman', 28),
(98, 'Jabalpur', 13),
(99, 'Indore', 13),
(100, 'Madhubani', 4),
(101, 'Darbhanga', 4),
(102, 'Gandhinagar', 7),
(103, 'Kharagpur', 28),
(104, 'Bangalore', 11),
(105, 'Tiruchirappalli', 23),
(106, 'Surathkal', 11),
(107, 'Vellore', 23),
(108, 'Manipal', 11),
(109, 'Phagwara', 20),
(110, 'Thanjavur', 23),
(111, 'Mesra', 10),
(112, 'Mirzapur', 26),
(113, 'Sri City', 1),
(114, 'Bhiwadi', 21),
(115, 'Neemrana', 21),
(116, 'Pune', 14),
(117, 'Gharuan', 20),
(118, 'Ratnagiri', 14),
(119, 'Rishikesh', 27),
(120, 'Sonipat', 8),
(121, 'Jhansi', 26),
(122, 'Tonk', 21),
(123, 'Coimbatore', 23),
-

(124, 'Palghar', 14),
(125, 'Mandi', 9),
(126, 'Bhilai', 5),
(127, 'Kanpur', 26),
(128, 'Pilani', 21),
(129, 'Jammu', 35),
(130, 'Kozhikode', 12),
(131, 'Wardha', 14),
(132, 'Amravati', 14),
(133, 'Vallabh Vidyanagar', 7),
(134, 'Udhampur', 35),
(135, 'Midnapore', 28),
(136, 'Rajkot', 7),
(137, 'Faridkot', 20),
(138, 'Salem', 23),
(139, 'Hassan', 11),
(140, 'Kakinada', 1),
(141, 'Vasad', 7);

3) Institute Table

```
INSERT INTO Institute (Institution_ID, Name, Tuition_Fees,  
Address, Website, Accreditation_Status, Established_Year, Type)  
VALUES  
(1, 'IIT Bombay', 200000, 'Mumbai, Maharashtra',  
'http://www.iitb.ac.in', TRUE, 1958, 'Technology'),  
(2, 'IIT Delhi', 200000, 'New Delhi', 'http://www.iitd.ac.in', TRUE,  
1961, 'Technology'),
```

-
- (3, 'IIT Madras', 200000, 'Chennai, Tamil Nadu',
'<http://www.iitm.ac.in>', TRUE, 1959, 'Technology'),
(4, 'IIT Kanpur', 200000, 'Kanpur, Uttar Pradesh',
'<http://www.iitk.ac.in>', TRUE, 1959, 'Technology'),
(5, 'IIT Kharagpur', 200000, 'Kharagpur, West Bengal',
'<http://www.iitkgp.ac.in>', TRUE, 1951, 'Technology'),
(6, 'IIM Ahmedabad', 250000, 'Ahmedabad, Gujarat',
'<http://www.iima.ac.in>', TRUE, 1961, 'Management'),
(7, 'IIM Bangalore', 250000, 'Bangalore, Karnataka',
'<http://www.iimb.ac.in>', TRUE, 1973, 'Management'),
(8, 'IIM Calcutta', 250000, 'Kolkata, West Bengal',
'<http://www.iimcal.ac.in>', TRUE, 1961, 'Management'),
(9, 'BITS Pilani', 150000, 'Pilani, Rajasthan',
'<http://www.bits-pilani.ac.in>', TRUE, 1964, 'Technology'),
(10, 'NIT Trichy', 120000, 'Tiruchirappalli, Tamil Nadu',
'<http://www.nitt.edu>', TRUE, 1971, 'Technology'),
(11, 'NIT Surathkal', 120000, 'Surathkal, Karnataka',
'<http://www.nitk.ac.in>', TRUE, 1960, 'Technology'),
(12, 'VIT Vellore', 200000, 'Vellore, Tamil Nadu',
'<http://www.vit.ac.in>', TRUE, 1984, 'Technology'),
(13, 'SRM Institute of Science and Technology', 150000, 'Chennai, Tamil Nadu', '<http://www.srmist.edu.in>', TRUE, 1985, 'Technology'),
(14, 'Manipal Institute of Technology', 120000, 'Manipal, Karnataka', '<http://www.manipal.edu>', TRUE, 1957, 'Technology'),
(15, 'LPU', 80000, 'Phagwara, Punjab', '<http://www.lpu.in>', TRUE, 2009, 'Technology'),
(16, 'Shiv Nadar University', 180000, 'Greater Noida, Uttar Pradesh', '<http://www.snu.edu.in>', TRUE, 2011, 'Technology'),
-

-
- (17, 'Delhi University', 50000, 'New Delhi', '<http://www.du.ac.in>', TRUE, 1922, 'Arts'),
(18, 'Jawaharlal Nehru University', 60000, 'New Delhi', '<http://www.jnu.ac.in>', TRUE, 1969, 'Arts'),
(19, 'Jamia Millia Islamia', 70000, 'New Delhi', '<http://www.jmi.ac.in>', TRUE, 1920, 'Arts'),
(20, 'Panjab University', 50000, 'Chandigarh', '<http://www.puchd.ac.in>', TRUE, 1882, 'Arts'),
(21, 'Tata Institute of Social Sciences', 70000, 'Mumbai, Maharashtra', '<http://www.tiss.edu>', TRUE, 1936, 'Social Sciences'),
(22, 'Indian Statistical Institute', 60000, 'Kolkata, West Bengal', '<http://www.isical.ac.in>', TRUE, 1931, 'Statistics'),
(23, 'National Institute of Fashion Technology', 120000, 'New Delhi', '<http://www.nift.ac.in>', TRUE, 1986, 'Fashion Design'),
(24, 'National Institute of Design', 120000, 'Ahmedabad, Gujarat', '<http://www.nid.edu>', TRUE, 1961, 'Design'),
(25, 'Amity University', 100000, 'Noida, Uttar Pradesh', '<http://www.amity.edu>', TRUE, 2005, 'Multidisciplinary'),
(26, 'Banaras Hindu University', 50000, 'Varanasi, Uttar Pradesh', '<http://www.bhu.ac.in>', TRUE, 1916, 'Arts'),
(27, 'University of Mumbai', 60000, 'Mumbai, Maharashtra', '<http://www.mu.ac.in>', TRUE, 1857, 'Multidisciplinary'),
(28, 'University of Chennai', 50000, 'Chennai, Tamil Nadu', '<http://www.unom.ac.in>', TRUE, 1857, 'Multidisciplinary'),
(29, 'SASTRA University', 120000, 'Thanjavur, Tamil Nadu', '<http://www.sastra.edu>', TRUE, 1984, 'Technology'),
(30, 'K L University', 80000, 'Vijayawada, Andhra Pradesh', '<http://www.kluniversity.in>', TRUE, 1980, 'Technology'),
-

-
- (31, 'Nirma University', 80000, 'Ahmedabad, Gujarat',
'<http://www.nirmauni.ac.in>', TRUE, 2003, 'Technology'),
(32, 'Birla Institute of Technology', 120000, 'Mesra, Jharkhand',
'<http://www.bitmesra.ac.in>', TRUE, 1955, 'Technology'),
(33, 'IIIT Bangalore', 120000, 'Bangalore, Karnataka',
'<http://www.iiitb.ac.in>', TRUE, 1999, 'Technology'),
(34, 'Krea University', 160000, 'Sri City, Andhra Pradesh',
'<http://www.krea.edu.in>', TRUE, 2018, 'Multidisciplinary'),
(35, 'Shivaji University', 40000, 'Kolhapur, Maharashtra',
'<http://www.unishivaji.ac.in>', TRUE, 1962, 'Multidisciplinary'),
(36, 'Lovely Professional University', 80000, 'Phagwara, Punjab',
'<http://www.lpu.in>', TRUE, 2009, 'Multidisciplinary'),
(37, 'Jadavpur University', 60000, 'Kolkata, West Bengal',
'<http://www.jaduniv.edu.in>', TRUE, 1955, 'Multidisciplinary'),
(38, 'GGS Indraprastha University', 50000, 'New Delhi',
'<http://www.ipu.ac.in>', TRUE, 1998, 'Multidisciplinary'),
(39, 'NIIT University', 70000, 'Neemrana, Rajasthan',
'<http://www.niituniversity.in>', TRUE, 2009, 'Technology'),
(40, 'PDPU', 60000, 'Gandhinagar, Gujarat',
'<http://www.pdpu.ac.in>', TRUE, 2007, 'Technology'),
(41, 'UPES', 80000, 'Dehradun, Uttarakhand',
'<http://www.upes.ac.in>', TRUE, 2003, 'Technology'),
(42, 'Bharati Vidyapeeth University', 70000, 'Pune, Maharashtra',
'<http://www.bvuniversity.edu.in>', TRUE, 1964, 'Multidisciplinary'),
(43, 'Delhi Technological University', 60000, 'New Delhi',
'<http://www.dtu.ac.in>', TRUE, 1941, 'Technology'),
(44, 'Netaji Subhas University of Technology', 60000, 'New Delhi',
'<http://www.nsut.ac.in>', TRUE, 1983, 'Technology'),
-

-
- (45, 'Mahatma Gandhi University', 50000, 'Kottayam, Kerala',
'<http://www.mgu.ac.in>', TRUE, 1983, 'Multidisciplinary'),
(46, 'Chandigarh University', 60000, 'Gharuan, Punjab',
'<http://www.cuchd.in>', TRUE, 2012, 'Multidisciplinary'),
(47, 'REVA University', 60000, 'Bangalore, Karnataka',
'<http://www.reva.edu.in>', TRUE, 2012, 'Technology'),
(48, 'Vellore Institute of Technology', 200000, 'Vellore, Tamil
Nadu', '<http://www.vit.ac.in>', TRUE, 1984, 'Technology'),
(49, 'M S Ramaiah University', 80000, 'Bangalore, Karnataka',
'<http://www.msruas.ac.in>', TRUE, 2013, 'Technology'),
(50, 'SRM Institute of Science and Technology', 150000, 'Chennai,
Tamil Nadu', '<http://www.srmuniv.ac.in>', TRUE, 1985,
'Technology'),
(51, 'SASTRA University', 120000, 'Thanjavur, Tamil Nadu',
'<http://www.sastra.edu.in>', TRUE, 1984, 'Technology'),
(52, 'Bharti Vidyapeeth Deemed University', 70000, 'Pune,
Maharashtra', '<http://www.bvdu.edu.in>', TRUE, 1996,
'Multidisciplinary'),
(53, 'Amrita Vishwa Vidyapeetham', 100000, 'Coimbatore, Tamil
Nadu', '<http://www.amrita.edu>', TRUE, 2003, 'Multidisciplinary'),
(54, 'Sikkim Manipal University', 70000, 'Gangtok, Sikkim',
'<http://www.smu.edu.in>', TRUE, 1995, 'Multidisciplinary'),
(55, 'University of Calcutta', 60000, 'Kolkata, West Bengal',
'<http://www.caluniv.ac.in>', TRUE, 1857, 'Multidisciplinary'),
(56, 'Jawaharlal Nehru Technological University', 50000,
'Hyderabad, Telangana', '<http://www.jntuh.ac.in>', TRUE, 1972,
'Technology'),
(57, 'Manipal University Jaipur', 70000, 'Jaipur, Rajasthan',
'<http://www.muj.manipal.edu>', TRUE, 2011, 'Technology'),
-

-
- (58, 'Shri Ram College of Commerce', 50000, 'New Delhi',
'<http://www.srcc.edu.in>', TRUE, 1926, 'Commerce'),
(59, 'National Law School of India University', 80000, 'Bangalore,
Karnataka', '<http://www.nls.ac.in>', TRUE, 1987, 'Law'),
(60, 'National Institute of Design', 120000, 'Ahmedabad, Gujarat',
'<http://www.nid.edu>', TRUE, 1961, 'Design'),
(61, 'National Institute of Fashion Technology', 120000, 'New
Delhi', '<http://www.nift.ac.in>', TRUE, 1986, 'Fashion'),
(62, 'Jamia Hamdard', 60000, 'New Delhi',
'<http://www.jamiahAMDARD.edu>', TRUE, 1989, 'Health Sciences'),
(63, 'Hamdard University', 60000, 'New Delhi',
'<http://www.hamdard.edu>', TRUE, 1989, 'Health Sciences'),
(64, 'Indian Institute of Technology, Mandi', 200000, 'Mandi,
Himachal Pradesh', '<http://www.iitmandi.ac.in>', TRUE, 2009,
'Technology'),
(65, 'Indian Institute of Technology, Bhilai', 200000, 'Bhilai,
Chhattisgarh', '<http://www.iitbhilai.ac.in>', TRUE, 2016,
'Technology'),
(66, 'Indian Institute of Technology, Jammu', 200000, 'Jammu,
Jammu and Kashmir', '<http://www.iitjammu.ac.in>', TRUE, 2016,
'Technology'),
(67, 'Indian Institute of Technology, Dhanbad', 200000, 'Dhanbad,
Jharkhand', '<http://www.iitism.ac.in>', TRUE, 1926, 'Technology'),
(68, 'Indian Institute of Management, Ranchi', 250000, 'Ranchi,
Jharkhand', '<http://www.iimranchi.ac.in>', TRUE, 2010,
'Management'),
(69, 'Indian Institute of Management, Lucknow', 250000,
'Lucknow, Uttar Pradesh', '<http://www.iiml.ac.in>', TRUE, 1984,
'Management'),

-
- (70, 'Indian Institute of Management, Kozhikode', 250000, 'Kozhikode, Kerala', '<http://www.iimk.ac.in>', TRUE, 1997, 'Management'),
- (71, 'Indian School of Business', 250000, 'Hyderabad, Telangana', '<http://www.isb.edu>', TRUE, 2001, 'Management'),
- (72, 'Karnataka State Open University', 40000, 'Mysuru, Karnataka', '<http://www.ksoumysore.karnataka.gov.in>', TRUE, 1996, 'Open University'),
- (73, 'Mahatma Gandhi Antarrashtriya Hindi Vishwavidyalaya', 60000, 'Wardha, Maharashtra', '<http://www.hindivishwa.org>', TRUE, 1997, 'Language'),
- (74, 'Nirma University', 80000, 'Ahmedabad, Gujarat', '<http://www.nirmauni.ac.in>', TRUE, 2003, 'Technology'),
- (75, 'Narsee Monjee Institute of Management Studies', 80000, 'Mumbai, Maharashtra', '<http://www.nmims.edu>', TRUE, 1994, 'Management'),
- (76, 'Ranchi University', 50000, 'Ranchi, Jharkhand', '<http://www.ranchiuniversity.ac.in>', TRUE, 1960, 'Multidisciplinary'),
- (77, 'University of Jammu', 60000, 'Jammu, Jammu and Kashmir', '<http://www.jammuuniversity.in>', TRUE, 1969, 'Multidisciplinary'),
- (78, 'Sardar Patel University', 60000, 'Vallabh Vidyanagar, Gujarat', '<http://www.spuvvn.edu>', TRUE, 1955, 'Multidisciplinary'),
- (79, 'Shivaji University', 40000, 'Kolhapur, Maharashtra', '<http://www.unishivaji.ac.in>', TRUE, 1962, 'Multidisciplinary'),
- (80, 'Tata Institute of Fundamental Research', 70000, 'Mumbai, Maharashtra', '<http://www.tifr.res.in>', TRUE, 1945, 'Research'),
- (81, 'University of Delhi', 50000, 'New Delhi', '<http://www.du.ac.in>', TRUE, 1922, 'Multidisciplinary'),

-
- (82, 'University of Kerala', 60000, 'Thiruvananthapuram, Kerala',
'<http://www.keralauniversity.ac.in>', TRUE, 1937,
'Multidisciplinary'),
- (83, 'Bharathiar University', 60000, 'Coimbatore, Tamil Nadu',
'<http://www.b-u.ac.in>', TRUE, 1982, 'Multidisciplinary'),
- (84, 'Vidyasagar University', 60000, 'Midnapore, West Bengal',
'<http://www.vidyasagar.ac.in>', TRUE, 1981, 'Multidisciplinary'),
- (85, 'Kochi University of Science and Technology', 70000, 'Kochi, Kerala', '<http://www.kust.edu.in>', TRUE, 1993, 'Technology'),
- (86, 'Kakatiya University', 50000, 'Warangal, Telangana',
'<http://www.kakatiya.ac.in>', TRUE, 1976, 'Multidisciplinary'),
- (87, 'Dr. B.R. Ambedkar University', 60000, 'Lucknow, Uttar Pradesh', '<http://www.dbrau.ac.in>', TRUE, 2007,
'Multidisciplinary'),
- (88, 'Saurashtra University', 60000, 'Rajkot, Gujarat',
'<http://www.saurashtrauniversity.edu>', TRUE, 1967,
'Multidisciplinary'),
- (89, 'Andhra University', 60000, 'Visakhapatnam, Andhra Pradesh',
'<http://www.andhrauniversity.edu.in>', TRUE, 1926,
'Multidisciplinary'),
- (90, 'Utkal University', 60000, 'Bhubaneswar, Odisha',
'<http://utkaluniversity.ac.in>', TRUE, 1943, 'Multidisciplinary'),
- (91, 'Baba Farid University of Health Sciences', 70000, 'Faridkot, Punjab', '<http://www.bfuhs.ac.in>', TRUE, 1998, 'Health Sciences'),
- (92, 'Gauhati University', 60000, 'Guwahati, Assam',
'<http://www.gauhati.ac.in>', TRUE, 1948, 'Multidisciplinary'),
- (93, 'Periyar University', 60000, 'Salem, Tamil Nadu',
'<http://www.periyaruniversity.ac.in>', TRUE, 1997,
'Multidisciplinary'),

(94, 'Tamil Nadu Agricultural University', 60000, 'Coimbatore, Tamil Nadu', 'http://www.tnau.ac.in', TRUE, 1971, 'Agricultural'),
(95, 'Jawaharlal Nehru Technological University, Kakinada', 60000, 'Kakinada, Andhra Pradesh', 'http://www.jntuk.edu.in', TRUE, 1946, 'Technology'),
(96, 'Guru Nanak Dev Engineering College', 70000, 'Ludhiana, Punjab', 'http://www.gndec.ac.in', TRUE, 1956, 'Engineering'),
(97, 'Punjab Engineering College', 70000, 'Chandigarh', 'http://www.pec.ac.in', TRUE, 1953, 'Engineering'),
(98, 'Sardar Vallabhbhai Patel Institute of Technology', 60000, 'Vasad, Gujarat', 'http://www.svpit.ac.in', TRUE, 2009, 'Technology'),
(99, 'Jawaharlal Nehru University, Hyderabad', 60000, 'Hyderabad, Telangana', 'http://www.jnuhyd.ac.in', TRUE, 2009, 'Multidisciplinary'),
(100, 'National Institute of Technical Teachers Training and Research', 60000, 'Bhopal, Madhya Pradesh', 'http://www.nittrbpl.ac.in', TRUE, 2002, 'Technical Teaching');

4) Employer Table

```
INSERT INTO Employer (Employer_ID, Employer_Name, Location, Salary_Benefits) VALUES  
(1, 'Tata Consultancy Services', 'Mumbai', 800000),  
(2, 'Infosys', 'Bengaluru', 750000),  
(3, 'Wipro', 'Bengaluru', 700000),  
(4, 'HCL Technologies', 'Noida', 720000),  
(5, 'Accenture', 'Gurgaon', 900000),  
(6, 'Cognizant', 'Chennai', 680000),
```

-
- (7, 'Tech Mahindra', 'Pune', 640000),
(8, 'Capgemini', 'Mumbai', 750000),
(9, 'IBM', 'Bengaluru', 850000),
(10, 'Oracle', 'Hyderabad', 900000),
(11, 'SAP', 'Gurgaon', 950000),
(12, 'Dell', 'Bengaluru', 780000),
(13, 'Intel', 'Hyderabad', 830000),
(14, 'Microsoft', 'Hyderabad', 1000000),
(15, 'Amazon', 'Bengaluru', 1200000),
(16, 'Google', 'Hyderabad', 1100000),
(17, 'Facebook', 'Hyderabad', 1150000),
(18, 'Adobe', 'Noida', 900000),
(19, 'Salesforce', 'Bengaluru', 950000),
(20, 'Cisco', 'Bengaluru', 880000),
(21, 'Uber', 'Bengaluru', 1000000),
(22, 'Ola', 'Bengaluru', 700000),
(23, 'Zomato', 'Gurgaon', 600000),
(24, 'Swiggy', 'Bengaluru', 650000),
(25, 'Paytm', 'Noida', 750000),
(26, 'Razorpay', 'Bengaluru', 700000),
(27, 'PhonePe', 'Bengaluru', 720000),
(28, 'Myntra', 'Bengaluru', 680000),
(29, 'Nykaa', 'Mumbai', 600000),
(30, 'Flipkart', 'Bengaluru', 950000),
(31, 'Snapdeal', 'Gurgaon', 500000),
(32, 'Tata Steel', 'Jamshedpur', 700000),
(33, 'Reliance Industries', 'Mumbai', 850000),
(34, 'Hindustan Unilever', 'Mumbai', 900000),
(35, 'ITC', 'Kolkata', 850000),
-

-
- (36, 'L&T', 'Mumbai', 800000),
(37, 'Godrej', 'Mumbai', 750000),
(38, 'Mahindra & Mahindra', 'Mumbai', 800000),
(39, 'Bajaj Auto', 'Pune', 750000),
(40, 'TVS Motor Company', 'Chennai', 680000),
(41, 'Hero MotoCorp', 'Gurgaon', 700000),
(42, 'Bharat Forge', 'Pune', 720000),
(43, 'Asian Paints', 'Mumbai', 740000),
(44, 'Maruti Suzuki', 'Gurgaon', 850000),
(45, 'Tata Motors', 'Pune', 800000),
(46, 'Nirma', 'Ahmedabad', 680000),
(47, 'P&G', 'Mumbai', 900000),
(48, 'Nestle', 'Mumbai', 850000),
(49, 'Coca-Cola', 'Bengaluru', 780000),
(50, 'PepsiCo', 'Hyderabad', 800000),
(51, 'Britannia', 'Mumbai', 700000),
(52, 'Hindustan Aeronautics', 'Bengaluru', 950000),
(53, 'ISRO', 'Bengaluru', 1000000),
(54, 'DRDO', 'Bengaluru', 900000),
(55, 'BHEL', 'Bhopal', 800000),
(56, 'NTPC', 'New Delhi', 850000),
(57, 'ONGC', 'Dehradun', 900000),
(58, 'GAIL', 'New Delhi', 750000),
(59, 'Cairn India', 'Rajasthan', 700000),
(60, 'Adani Group', 'Ahmedabad', 850000),
(61, 'JSW Steel', 'Bengaluru', 800000),
(62, 'Steel Authority of India', 'New Delhi', 720000),
(63, 'Indian Oil Corporation', 'New Delhi', 750000),
(64, 'Hindustan Zinc', 'Udaipur', 680000),
-

-
- (65, 'Marico', 'Mumbai', 700000),
(66, 'ITC Hotels', 'Kolkata', 750000),
(67, 'KFC', 'New Delhi', 600000),
(68, 'McDonald's', 'Mumbai', 650000),
(69, 'Domino's Pizza', 'Bengaluru', 620000),
(70, 'Starbucks', 'Mumbai', 650000),
(71, 'Taj Hotels', 'Mumbai', 800000),
(72, 'Oyo Rooms', 'Gurgaon', 700000),
(73, 'Zomato Gold', 'Bengaluru', 600000),
(74, 'MakeMyTrip', 'Gurgaon', 650000),
(75, 'Cleartrip', 'Mumbai', 580000),
(76, 'GoAir', 'Mumbai', 620000),
(77, 'IndiGo', 'Gurgaon', 680000),
(78, 'Air India', 'Mumbai', 750000),
(79, 'SpiceJet', 'Delhi', 600000),
(80, 'Vistara', 'Delhi', 700000),
(81, 'Bharti Airtel', 'New Delhi', 800000),
(82, 'Reliance Jio', 'Mumbai', 850000),
(83, 'Vodafone', 'Mumbai', 700000),
(84, 'BSNL', 'Mumbai', 600000),
(85, 'MTNL', 'Mumbai', 500000),
(86, 'Google Pay', 'Hyderabad', 720000),
(87, 'PhonePe', 'Bengaluru', 700000),
(88, 'FreeCharge', 'Gurgaon', 650000),
(89, 'PayPal', 'Hyderabad', 800000),
(90, 'PayU', 'Gurgaon', 700000),
(91, 'Razorpay', 'Bengaluru', 720000),
(92, 'Cure.fit', 'Bengaluru', 650000),
(93, 'PharmEasy', 'Mumbai', 700000),
-

(94, '1mg', 'Bengaluru', 680000),
(95, 'Netmeds', 'Hyderabad', 640000),
(96, 'Zomato', 'Gurgaon', 600000),
(97, 'Swiggy', 'Bengaluru', 650000),
(98, 'Urban Company', 'Gurgaon', 680000),
(99, 'Practo', 'Bengaluru', 700000),
(100, 'Dunzo', 'Bengaluru', 600000);

5) Health_Facility Table

INSERT INTO Health_Facility (Facility_ID, Name, Facility_Type, Bed_Capacity) VALUES
(1, 'All India Institute of Medical Sciences', 'Hospital', 800),
(2, 'Post Graduate Institute of Medical Education and Research', 'Hospital', 1500),
(3, 'Tata Memorial Hospital', 'Cancer Hospital', 600),
(4, 'Fortis Hospital', 'Multi-Specialty Hospital', 200),
(5, 'Apollo Hospitals', 'Multi-Specialty Hospital', 400),
(6, 'Max Super Specialty Hospital', 'Multi-Specialty Hospital', 300),
(7, 'Narayana Health', 'Multi-Specialty Hospital', 1000),
(8, 'Medanta – The Medicity', 'Multi-Specialty Hospital', 800),
(9, 'Kokilaben Dhirubhai Ambani Hospital', 'Multi-Specialty Hospital', 300),
(10, 'Manipal Hospital', 'Multi-Specialty Hospital', 500),
(11, 'Lilavati Hospital', 'Multi-Specialty Hospital', 350),
(12, 'Sri Ramachandra Medical Centre', 'Hospital', 700),
(13, 'Sankara Nethralaya', 'Eye Hospital', 250),
(14, 'P.D. Hinduja Hospital', 'Multi-Specialty Hospital', 300),

-
- (15, 'Care Hospital', 'Multi-Specialty Hospital', 350),
 - (16, 'Jaypee Hospital', 'Multi-Specialty Hospital', 200),
 - (17, 'Sitaram Bhartia Institute of Science and Research', 'Hospital', 100),
 - (18, 'HCG Cancer Centre', 'Cancer Hospital', 150),
 - (19, 'Aster CMI Hospital', 'Multi-Specialty Hospital', 250),
 - (20, 'Asian Institute of Medical Sciences', 'Multi-Specialty Hospital', 400),
 - (21, 'Narayana Institute of Cardiac Sciences', 'Heart Hospital', 200),
 - (22, 'Apollo Spectra Hospitals', 'Surgical Hospital', 150),
 - (23, 'Sahyadri Hospital', 'Multi-Specialty Hospital', 300),
 - (24, 'MGM Healthcare', 'Multi-Specialty Hospital', 400),
 - (25, 'MediHope Super Specialty Hospital', 'Multi-Specialty Hospital', 200),
 - (26, 'BLK Super Specialty Hospital', 'Multi-Specialty Hospital', 300),
 - (27, 'Sri Aurobindo Institute of Medical Sciences', 'Hospital', 400),
 - (28, 'Sankalp Hospital', 'Multi-Specialty Hospital', 150),
 - (29, 'Jaslok Hospital', 'Multi-Specialty Hospital', 600),
 - (30, 'Indraprastha Apollo Hospital', 'Multi-Specialty Hospital', 500),
 - (31, 'Hiranandani Hospital', 'Multi-Specialty Hospital', 300),
 - (32, 'Vijaya Medical Centre', 'Hospital', 250),
 - (33, 'Reddy Hospitals', 'Multi-Specialty Hospital', 150),
 - (34, 'Fortis Escorts Heart Institute', 'Heart Hospital', 250),
 - (35, 'Yashoda Hospitals', 'Multi-Specialty Hospital', 400),
 - (36, 'Zydus Hospitals', 'Multi-Specialty Hospital', 200),
 - (37, 'Vivekananda Hospital', 'Multi-Specialty Hospital', 300),

-
- (38, 'St. John's Medical College Hospital', 'Hospital', 500),
(39, 'Sanjivani Hospital', 'Multi-Specialty Hospital', 150),
(40, 'Bansal Hospital', 'Multi-Specialty Hospital', 200),
(41, 'Max Healthcare', 'Multi-Specialty Hospital', 600),
(42, 'Wockhardt Hospitals', 'Multi-Specialty Hospital', 350),
(43, 'Manipal Comprehensive Cancer Care', 'Cancer Hospital', 150),
(44, 'Heritage Hospital', 'Multi-Specialty Hospital', 100),
(45, 'Lifecare Hospital', 'Multi-Specialty Hospital', 200),
(46, 'Oasis Hospital', 'Multi-Specialty Hospital', 300),
(47, 'KIMS Hospital', 'Multi-Specialty Hospital', 500),
(48, 'Fortis La Femme Hospital', 'Women's Hospital', 200),
(49, 'St. Thomas Hospital', 'Hospital', 400),
(50, 'Shalby Hospitals', 'Multi-Specialty Hospital', 350),
(51, 'Health City', 'Multi-Specialty Hospital', 300),
(52, 'Cloudnine Hospital', 'Women's Hospital', 150),
(53, 'Sanghvi Hospital', 'Multi-Specialty Hospital', 200),
(54, 'Mindsprings Hospital', 'Psychiatric Hospital', 100),
(55, 'Sanjeevani Hospital', 'Multi-Specialty Hospital', 300),
(56, 'Swastik Hospital', 'Multi-Specialty Hospital', 150),
(57, 'Puspa Hospital', 'Multi-Specialty Hospital', 250),
(58, 'LifeCare Hospitals', 'Multi-Specialty Hospital', 100),
(59, 'Shanti Nursing Home', 'Nursing Home', 50),
(60, 'Dr. Lal PathLabs', 'Diagnostics Center', 0),
(61, 'Max HealthCare', 'Multi-Specialty Hospital', 600),
(62, 'Srinivas Hospital', 'Multi-Specialty Hospital', 200),
(63, 'Star Hospitals', 'Multi-Specialty Hospital', 150),
(64, 'Siddharth Hospital', 'Multi-Specialty Hospital', 250),
(65, 'Care Clinic', 'Clinic', 10),
-

-
- (66, 'Sankalp Nursing Home', 'Nursing Home', 30),
(67, 'Karnataka Hospital', 'Multi-Specialty Hospital', 400),
(68, 'Shraddha Hospital', 'Multi-Specialty Hospital', 300),
(69, 'Prashanth Hospital', 'Multi-Specialty Hospital', 200),
(70, 'Radiant Hospital', 'Multi-Specialty Hospital', 100),
(71, 'Sarvodaya Hospital', 'Multi-Specialty Hospital', 300),
(72, 'Venkateshwara Hospital', 'Multi-Specialty Hospital', 500),
(73, 'Kasturba Hospital', 'Hospital', 600),
(74, 'Karnavati Hospital', 'Multi-Specialty Hospital', 250),
(75, 'Muktai Hospital', 'Multi-Specialty Hospital', 300),
(76, 'Vatsalya Hospital', 'Multi-Specialty Hospital', 150),
(77, 'Nandini Hospital', 'Multi-Specialty Hospital', 200),
(78, 'Gujarat Hospital', 'Multi-Specialty Hospital', 400),
(79, 'Patan Hospital', 'Multi-Specialty Hospital', 100),
(80, 'Sushrut Hospital', 'Multi-Specialty Hospital', 300),
(81, 'Dhanvantri Hospital', 'Multi-Specialty Hospital', 600),
(82, 'Universal Hospital', 'Multi-Specialty Hospital', 250),
(83, 'Yashoda Cancer Institute', 'Cancer Hospital', 100),
(84, 'Jeevan Hospital', 'Multi-Specialty Hospital', 300),
(85, 'Om Hospital', 'Multi-Specialty Hospital', 500),
(86, 'Gleneagles Global Health City', 'Multi-Specialty Hospital', 700),
(87, 'Pyramid Hospital', 'Multi-Specialty Hospital', 200),
(88, 'Sai Hospital', 'Multi-Specialty Hospital', 100),
(89, 'Shubham Hospital', 'Multi-Specialty Hospital', 200),
(90, 'Srinivasa Hospital', 'Multi-Specialty Hospital', 300),
(91, 'MediHelp Hospital', 'Multi-Specialty Hospital', 500),
(92, 'Eden Hospital', 'Multi-Specialty Hospital', 600),
(93, 'Medicure Hospital', 'Multi-Specialty Hospital', 350),
-

(94, 'Columbia Asia Hospital', 'Multi-Specialty Hospital', 400),
(95, 'Pushpanjali Hospital', 'Multi-Specialty Hospital', 250),
(96, 'Chaitanya Hospital', 'Multi-Specialty Hospital', 100),
(97, 'Arogyam Hospital', 'Multi-Specialty Hospital', 300),
(98, 'Motherhood Hospital', 'Women's Hospital', 200),
(99, 'Sunshine Hospital', 'Multi-Specialty Hospital', 500),
(100, 'Jaypee Hospital', 'Multi-Specialty Hospital', 400);

6) Government_Policy Table

INSERT INTO Government_Policy (Policy_ID, Policy_Name, Policy_Type, State_ID) VALUES
(1, 'Migration Support Scheme', 'Migration', 1),
(2, 'Interstate Migrant Workers Act', 'Migration', 3),
(3, 'Bihar Migrant Labour Welfare Program', 'Migration', 4),
(4, 'Chhattisgarh Migrant Assistance', 'Migration', 5),
(5, 'Goa Migrant Worker Act', 'Migration', 6),
(6, 'Gujarat Migrant Welfare Board', 'Migration', 7),
(7, 'Haryana Migrant Worker Program', 'Migration', 8),
(8, 'Himachal Pradesh Migrant Assistance', 'Migration', 9),
(9, 'Jharkhand Migrant Support', 'Migration', 10),
(10, 'Karnataka Migrant Worker Policy', 'Migration', 11),
(11, 'Kerala Migrant Welfare Program', 'Migration', 12),
(12, 'Madhya Pradesh Migration Assistance', 'Migration', 13),
(13, 'Maharashtra Migrant Worker Act', 'Migration', 14),
(14, 'Odisha Migrant Labour Welfare Scheme', 'Migration', 19),
(15, 'Punjab Migrant Support Program', 'Migration', 20),
(16, 'Rajasthan Migrant Worker Program', 'Migration', 21),
(17, 'Tamil Nadu Migrant Assistance', 'Migration', 23),

-
- (18, 'Telangana Migrant Welfare Act', 'Migration', 24),
(19, 'West Bengal Migrant Assistance', 'Migration', 28),
(20, 'Delhi Migration Support Policy', 'Migration', 33),
(21, 'Andhra Pradesh Job Creation Scheme', 'Job', 1),
(22, 'Arunachal Pradesh Employment Program', 'Job', 2),
(23, 'Assam Job Guarantee Act', 'Job', 3),
(24, 'Bihar Skill Development Mission', 'Job', 4),
(25, 'Chhattisgarh Employment Opportunity Scheme', 'Job', 5),
(26, 'Goa Youth Employment Program', 'Job', 6),
(27, 'Gujarat Industrial Job Scheme', 'Job', 7),
(28, 'Haryana Skill Development', 'Job', 8),
(29, 'Himachal Pradesh Job Assurance Program', 'Job', 9),
(30, 'Jharkhand Job Training Scheme', 'Job', 10),
(31, 'Karnataka Employment Guarantee', 'Job', 11),
(32, 'Kerala Startup Job Program', 'Job', 12),
(33, 'Madhya Pradesh Job Creation Mission', 'Job', 13),
(34, 'Maharashtra Rural Employment Scheme', 'Job', 14),
(35, 'Punjab Employment Guarantee', 'Job', 20),
(36, 'Rajasthan Job Training Initiative', 'Job', 21),
(37, 'Sikkim Youth Employment Mission', 'Job', 22),
(38, 'Tamil Nadu Skill Development', 'Job', 23),
(39, 'Uttar Pradesh Job Support Scheme', 'Job', 26),
(40, 'West Bengal Employment Mission', 'Job', 28),
(41, 'Andhra Pradesh Free Education Scheme', 'Education', 1),
(42, 'Assam Scholarship Program', 'Education', 3),
(43, 'Bihar Free Tuition Program', 'Education', 4),
(44, 'Chhattisgarh Education Loan Scheme', 'Education', 5),
(45, 'Goa Student Support Program', 'Education', 6),
(46, 'Gujarat Educational Aid Scheme', 'Education', 7),
-

-
- (47, 'Haryana Scholarship Program', 'Education', 8),
(48, 'Himachal Pradesh Free Education Act', 'Education', 9),
(49, 'Jharkhand Tuition Fee Waiver', 'Education', 10),
(50, 'Karnataka Merit Scholarship', 'Education', 11),
(51, 'Kerala Student Welfare Scheme', 'Education', 12),
(52, 'Madhya Pradesh Scholarship Scheme', 'Education', 13),
(53, 'Maharashtra Education Aid Program', 'Education', 14),
(54, 'Manipur Free Education Initiative', 'Education', 15),
(55, 'Nagaland Scholarship Program', 'Education', 18),
(56, 'Odisha Educational Support Scheme', 'Education', 19),
(57, 'Punjab Student Welfare Act', 'Education', 20),
(58, 'Rajasthan School Support Scheme', 'Education', 21),
(59, 'Tamil Nadu Education Assistance', 'Education', 23),
(60, 'Uttar Pradesh Merit Scholarship', 'Education', 26),
(61, 'Andhra Pradesh Health Insurance Scheme', 'Health', 1),
(62, 'Arunachal Pradesh Medical Support', 'Health', 2),
(63, 'Assam Health Protection Scheme', 'Health', 3),
(64, 'Bihar Health Assurance Plan', 'Health', 4),
(65, 'Chhattisgarh Rural Health Initiative', 'Health', 5),
(66, 'Goa Free Health Check-up Program', 'Health', 6),
(67, 'Gujarat Medical Assistance Scheme', 'Health', 7),
(68, 'Haryana Healthcare Program', 'Health', 8),
(69, 'Himachal Pradesh Health Initiative', 'Health', 9),
(70, 'Jharkhand Rural Health Scheme', 'Health', 10),
(71, 'Karnataka Health Insurance Program', 'Health', 11),
(72, 'Kerala Healthcare Assistance', 'Health', 12),
(73, 'Madhya Pradesh Health Coverage Program', 'Health', 13),
(74, 'Maharashtra Health Insurance Act', 'Health', 14),
(75, 'Odisha Health and Wellness Program', 'Health', 19),
-

-
- (76, 'Punjab Health Assurance Scheme', 'Health', 20),
(77, 'Rajasthan Health Welfare Program', 'Health', 21),
(78, 'Tamil Nadu Healthcare Mission', 'Health', 23),
(79, 'Uttarakhand Health Insurance Plan', 'Health', 27),
(80, 'West Bengal Medical Support Scheme', 'Health', 28),
(81, 'Delhi Migrant Welfare Act', 'Migration', 33),
(82, 'Jammu and Kashmir Employment Guarantee', 'Job', 35),
(83, 'Tripura Free Education Program', 'Education', 25),
(84, 'Ladakh Health Support Plan', 'Health', 36),
(85, 'Puducherry Skill Development Scheme', 'Job', 34),
(86, 'Lakshadweep Student Support Program', 'Education', 32),
(87, 'Dadra and Nagar Haveli Job Opportunity Scheme', 'Job', 31),
(88, 'Andaman and Nicobar Health Program', 'Health', 29),
(89, 'Chandigarh Migrant Assistance Program', 'Migration', 30),
(90, 'Sikkim Education for All Initiative', 'Education', 22),
(91, 'Mizoram Employment Assistance', 'Job', 17),
(92, 'Manipur Education Support Scheme', 'Education', 15),
(93, 'Nagaland Healthcare Mission', 'Health', 18),
(94, 'Meghalaya Job Training Program', 'Job', 16),
(95, 'Kerala Rural Education Fund', 'Education', 12),
(96, 'Punjab Migration Assistance', 'Migration', 20),
(97, 'Odisha Job Opportunity Scheme', 'Job', 19),
(98, 'Tamil Nadu Rural Health Program', 'Health', 23),
(99, 'Uttar Pradesh Student Loan Support', 'Education', 26),
(100, 'West Bengal Skill Development Initiative', 'Job', 28);

7) Climate Table

```
INSERT INTO Climate (City_ID, Policy_ID, Average_Temp,  
Climate_Type) VALUES  
(1, 1, 25.0, 'Tropical'),  
(1, 2, 26.5, 'Tropical'),  
(1, 3, 24.0, 'Tropical'),  
(2, 4, 30.0, 'Tropical'),  
(2, 5, 31.0, 'Tropical'),  
(2, 6, 28.5, 'Tropical'),  
(3, 7, 14.0, 'Temperate'),  
(3, 8, 15.0, 'Temperate'),  
(3, 9, 13.5, 'Temperate'),  
(4, 10, 20.0, 'Temperate'),  
(4, 11, 21.0, 'Temperate'),  
(4, 12, 19.5, 'Temperate'),  
(5, 13, 8.0, 'Cold'),  
(5, 14, 9.0, 'Cold'),  
(5, 15, 7.5, 'Cold'),  
(6, 16, 27.0, 'Tropical'),  
(6, 17, 26.5, 'Tropical'),  
(6, 18, 28.5, 'Tropical'),  
(7, 19, 15.0, 'Mediterranean'),  
(7, 20, 16.5, 'Mediterranean'),  
(7, 21, 14.5, 'Mediterranean'),  
(8, 22, 12.0, 'Cold'),  
(8, 23, 11.0, 'Cold'),  
(8, 24, 10.5, 'Cold'),  
(9, 25, 22.0, 'Tropical'),  
(9, 26, 21.5, 'Tropical'),
```

(9, 27, 23.0, 'Tropical'),
(10, 28, 18.0, 'Temperate'),
(10, 29, 19.0, 'Temperate'),
(10, 30, 17.5, 'Temperate'),
(11, 31, 29.0, 'Tropical'),
(11, 32, 28.0, 'Tropical'),
(11, 33, 30.0, 'Tropical'),
(12, 34, 9.5, 'Cold'),
(12, 35, 8.5, 'Cold'),
(12, 36, 10.0, 'Cold'),
(13, 37, 14.0, 'Temperate'),
(13, 38, 15.5, 'Temperate'),
(13, 39, 13.0, 'Temperate'),
(14, 40, 31.0, 'Tropical'),
(14, 41, 32.0, 'Tropical'),
(14, 42, 30.5, 'Tropical'),
(15, 43, 11.0, 'Cold'),
(15, 44, 12.5, 'Cold'),
(15, 45, 10.0, 'Cold'),
(16, 46, 26.0, 'Tropical'),
(16, 47, 25.0, 'Tropical'),
(16, 48, 27.5, 'Tropical'),
(17, 49, 8.0, 'Cold'),
(17, 50, 9.5, 'Cold'),
(17, 51, 7.5, 'Cold'),
(18, 52, 19.0, 'Temperate'),
(18, 53, 18.5, 'Temperate'),
(18, 54, 20.0, 'Temperate'),
(19, 55, 32.0, 'Tropical'),

(19, 56, 31.5, 'Tropical'),
(19, 57, 30.0, 'Tropical'),
(20, 58, 14.5, 'Mediterranean'),
(20, 59, 15.0, 'Mediterranean'),
(20, 60, 13.5, 'Mediterranean'),
(21, 61, 12.0, 'Cold'),
(21, 62, 10.0, 'Cold'),
(21, 63, 11.0, 'Cold'),
(22, 64, 28.0, 'Tropical'),
(22, 65, 29.5, 'Tropical'),
(22, 66, 27.0, 'Tropical'),
(23, 67, 17.0, 'Temperate'),
(23, 68, 16.5, 'Temperate'),
(23, 69, 18.5, 'Temperate'),
(24, 70, 31.0, 'Tropical'),
(24, 71, 32.5, 'Tropical'),
(24, 72, 30.5, 'Tropical'),
(25, 73, 8.0, 'Cold'),
(25, 74, 9.0, 'Cold'),
(25, 75, 7.0, 'Cold'),
(26, 76, 15.0, 'Mediterranean'),
(26, 77, 14.0, 'Mediterranean'),
(26, 78, 16.0, 'Mediterranean'),
(27, 79, 22.0, 'Tropical'),
(27, 80, 21.0, 'Tropical'),
(27, 81, 23.0, 'Tropical'),
(28, 82, 13.5, 'Temperate'),
(28, 83, 14.0, 'Temperate'),
(28, 84, 12.0, 'Temperate'),

(29, 85, 30.0, 'Tropical'),
(29, 86, 29.0, 'Tropical'),
(29, 87, 31.0, 'Tropical'),
(30, 88, 9.0, 'Cold'),
(30, 89, 8.5, 'Cold'),
(30, 90, 10.0, 'Cold'),
(31, 91, 17.0, 'Temperate'),
(31, 92, 18.0, 'Temperate'),
(31, 93, 16.5, 'Temperate'),
(32, 94, 26.0, 'Tropical'),
(32, 95, 25.5, 'Tropical'),
(32, 96, 27.5, 'Tropical'),
(33, 97, 12.0, 'Mediterranean'),
(33, 98, 11.0, 'Mediterranean'),
(33, 99, 13.0, 'Mediterranean'),
(34, 100, 30.0, 'Tropical');

8) Youngster Table

```
INSERT INTO Youngster (Youngster_ID, name, email, gender,  
Education_ID, Employment_ID, place_of_origin, Date_Of_Birth)  
VALUES  
(1, 'Aarav Sharma', 'aarav.sharma@gmail.com', 'Male', 12, 1,  
'Delhi', '1998-01-15'),  
(2, 'Ananya Gupta', 'ananya.gupta@gmail.com', 'Female', 23, 2,  
'Mumbai', '1999-02-25'),  
(3, 'Vihaan Kumar', 'vihaan.kumar@gmail.com', 'Male', 34, 3,  
'Bangalore', '2000-03-05'),  
(4, 'Aanya Verma', 'aanya.verma@gmail.com', 'Female', 45, 4,  
'Chennai', '1997-04-10'),
```

-
- (5, 'Arjun Reddy', 'arjun.reddy@gmail.com', 'Male', 56, 5, 'Hyderabad', '2001-05-22'),
(6, 'Pooja Singh', 'pooja.singh@gmail.com', 'Female', 67, 6, 'Kolkata', '1995-06-30'),
(7, 'Kabir Khan', 'kabir.khan@gmail.com', 'Male', 78, 7, 'Ahmedabad', '1998-07-15'),
(8, 'Diya Mehta', 'diya.mehta@gmail.com', 'Female', 89, 8, 'Surat', '1999-08-12'),
(9, 'Rohan Patel', 'rohan.patel@gmail.com', 'Male', 92, 9, 'Jaipur', '2000-09-09'),
(10, 'Meera Nair', 'meera.nair@gmail.com', 'Female', 11, 10, 'Pune', '1996-10-01'),
(11, 'Sai Kumar', 'sai.kumar@gmail.com', 'Male', 22, 11, 'Coimbatore', '1994-11-11'),
(12, 'Nisha Yadav', 'nisha.yadav@gmail.com', 'Female', 33, 12, 'Lucknow', '1998-12-21'),
(13, 'Ishaan Joshi', 'ishaan.joshi@gmail.com', 'Male', 44, 13, 'Kochi', '2001-01-03'),
(14, 'Sneha Desai', 'sneha.desai@gmail.com', 'Female', 55, 14, 'Visakhapatnam', '1997-02-17'),
(15, 'Akash Bansal', 'akash.bansal@gmail.com', 'Male', 66, 15, 'Nagpur', '2000-03-27'),
(16, 'Tara Rani', 'tara.rani@gmail.com', 'Female', 77, 16, 'Vadodara', '1999-04-30'),
(17, 'Karan Singh', 'karan.singh@gmail.com', 'Male', 88, 17, 'Indore', '1995-05-22'),
(18, 'Aditi Gupta', 'aditi.gupta@gmail.com', 'Female', 99, 18, 'Rajkot', '1998-06-15'),

-
- (19, 'Omkar Patil', 'omkar.patil@gmail.com', 'Male', 10, 19, 'Nashik', '2000-07-01'),
(20, 'Riya Sharma', 'riya.sharma@gmail.com', 'Female', 21, 20, 'Faridabad', '1997-08-12'),
(21, 'Aravind Iyer', 'aravind.iyer@gmail.com', 'Male', 32, 21, 'Mysuru', '1996-09-25'),
(22, 'Vaishali Yadav', 'vaishali.yadav@gmail.com', 'Female', 43, 22, 'Bhopal', '1999-10-14'),
(23, 'Devendra Rao', 'devendra.rao@gmail.com', 'Male', 54, 23, 'Thane', '1998-11-30'),
(24, 'Simran Khanna', 'simran.khanna@gmail.com', 'Female', 65, 24, 'Jodhpur', '1994-12-05'),
(25, 'Nitin Malhotra', 'nitin.malhotra@gmail.com', 'Male', 76, 25, 'Dehradun', '1999-01-20'),
(26, 'Preeti Chawla', 'preeti.chawla@gmail.com', 'Female', 87, 26, 'Agra', '1996-02-10'),
(27, 'Ravi Kumar', 'ravi.kumar@gmail.com', 'Male', 98, 27, 'Raipur', '2000-03-15'),
(28, 'Tanvi Jain', 'tanvi.jain@gmail.com', 'Female', 9, 28, 'Mangalore', '1995-04-18'),
(29, 'Harsh Mehta', 'harsh.mehta@gmail.com', 'Male', 20, 29, 'Guwahati', '2002-05-25'),
(30, 'Sonal Gupta', 'sonal.gupta@gmail.com', 'Female', 31, 30, 'Chandigarh', '1998-06-10'),
(31, 'Neeraj Gupta', 'neeraj.gupta@gmail.com', 'Male', 42, 31, 'Srinagar', '1997-07-01'),
(32, 'Aastha Shah', 'aastha.shah@gmail.com', 'Female', 53, 32, 'Bhavnagar', '1999-08-15'),

-
- (33, 'Mohan Tiwari', 'mohan.tiwari@gmail.com', 'Male', 64, 33, 'Bhubaneswar', '1996-09-10'),
- (34, 'Sonali Roy', 'sonali.roy@gmail.com', 'Female', 75, 34, 'Patna', '1998-10-20'),
- (35, 'Ajay Joshi', 'ajay.joshi@gmail.com', 'Male', 86, 35, 'Navi Mumbai', '1995-11-05'),
- (36, 'Pinky Sharma', 'pinky.sharma@gmail.com', 'Female', 97, 36, 'Chennai', '2001-12-25'),
- (37, 'Rishabh Sharma', 'rishabh.sharma@gmail.com', 'Male', 8, 37, 'Noida', '1998-01-30'),
- (38, 'Tanya Verma', 'tanya.verma@gmail.com', 'Female', 19, 38, 'Lucknow', '1999-02-15'),
- (39, 'Deepak Singh', 'deepak.singh@gmail.com', 'Male', 30, 39, 'Ghaziabad', '1997-03-05'),
- (40, 'Neha Bansal', 'neha.bansal@gmail.com', 'Female', 41, 40, 'Faridabad', '1996-04-22'),
- (41, 'Vivek Reddy', 'vivek.reddy@gmail.com', 'Male', 52, 41, 'Gurgaon', '2000-05-20'),
- (42, 'Pragya Kapoor', 'pragya.kapoor@gmail.com', 'Female', 63, 42, 'Meerut', '1995-06-25'),
- (43, 'Sidharth Kumar', 'sidharth.kumar@gmail.com', 'Male', 74, 43, 'Indore', '1998-07-15'),
- (44, 'Riya Mehta', 'riya.mehta@gmail.com', 'Female', 85, 44, 'Kochi', '1999-08-30'),
- (45, 'Aakash Iyer', 'akash.iyer@gmail.com', 'Male', 96, 45, 'Nashik', '1996-09-05'),
- (46, 'Deepika Agarwal', 'deepika.agarwal@gmail.com', 'Female', 7, 46, 'Jodhpur', '1999-10-11'),

-
- (47, 'Rahul Sharma', 'raahul.sharma@gmail.com', 'Male', 18, 47, 'Jaipur', '1998-11-20'),
- (48, 'Sakshi Singh', 'saakshi.singh@gmail.com', 'Female', 29, 48, 'Surat', '1997-12-01'),
- (49, 'Karan Yadav', 'karan.yadav@gmail.com', 'Male', 40, 49, 'Delhi', '1995-01-15'),
- (50, 'Megha Nair', 'megha.nair@gmail.com', 'Female', 51, 50, 'Bangalore', '1999-02-20'),
- (51, 'Shivam Patel', 'shivam.patel@gmail.com', 'Male', 62, 51, 'Ahmedabad', '1996-03-22'),
- (52, 'Aditi Verma', 'aditi.verma@gmail.com', 'Female', 73, 52, 'Pune', '1998-04-25'),
- (53, 'Manoj Kumar', 'manoj.kumar@gmail.com', 'Male', 84, 53, 'Mumbai', '2000-05-30'),
- (54, 'Priya Jain', 'priya.jain@gmail.com', 'Female', 95, 54, 'Kolkata', '1997-06-28'),
- (55, 'Vikas Reddy', 'vikas.reddy@gmail.com', 'Male', 6, 55, 'Chennai', '1998-07-10'),
- (56, 'Simran Kaur', 'simran.kaur@gmail.com', 'Female', 17, 56, 'Coimbatore', '1996-08-15'),
- (57, 'Kunal Gupta', 'kunal.gupta@gmail.com', 'Male', 28, 57, 'Hyderabad', '1999-09-12'),
- (58, 'Neha Rani', 'neha.rani@gmail.com', 'Female', 39, 58, 'Nagpur', '2000-10-19'),
- (59, 'Ravi Kumar', 'ravii.kumar@gmail.com', 'Male', 50, 59, 'Ahmedabad', '1998-11-25'),
- (60, 'Shreya Singh', 'shreya.singh@gmail.com', 'Female', 61, 60, 'Delhi', '1996-12-30'),

-
- (61, 'Amit Desai', 'amit.desai@gmail.com', 'Male', 72, 61, 'Jaipur', '1995-01-11'),
(62, 'Poonam Thakur', 'poonam.thakur@gmail.com', 'Female', 83, 62, 'Faridabad', '1998-02-01'),
(63, 'Tarun Verma', 'tarun.verma@gmail.com', 'Male', 94, 63, 'Mumbai', '1999-03-15'),
(64, 'Ritika Agarwal', 'ritika.agarwal@gmail.com', 'Female', 5, 64, 'Lucknow', '1997-04-20'),
(65, 'Rajesh Singh', 'rajesh.singh@gmail.com', 'Male', 16, 65, 'Raipur', '1999-05-30'),
(66, 'Suman Saini', 'suman.saini@gmail.com', 'Female', 27, 66, 'Bhopal', '2001-06-15'),
(67, 'Nikhil Yadav', 'nikhil.yadav@gmail.com', 'Male', 38, 67, 'Indore', '1995-07-01'),
(68, 'Kajal Sharma', 'kajal.sharma@gmail.com', 'Female', 49, 68, 'Pune', '1996-08-20'),
(69, 'Apoorva Kaur', 'apoorva.kaur@gmail.com', 'Female', 60, 69, 'Nagpur', '1998-09-11'),
(70, 'Gaurav Yadav', 'gaurav.yadav@gmail.com', 'Male', 71, 70, 'Delhi', '2000-10-12'),
(71, 'Shivani Sharma', 'shivani.sharma@gmail.com', 'Female', 82, 71, 'Chennai', '1996-11-15'),
(72, 'Akhil Reddy', 'akhil.reddy@gmail.com', 'Male', 93, 72, 'Kolkata', '1998-12-01'),
(73, 'Naina Gupta', 'naina.gupta@gmail.com', 'Female', 4, 73, 'Bangalore', '1995-01-21'),
(74, 'Rishab Singh', 'rishab.singh@gmail.com', 'Male', 15, 74, 'Hyderabad', '1999-02-18'),

-
- (75, 'Anjali Yadav', 'anjali.yadav@gmail.com', 'Female', 26, 75, 'Ahmedabad', '1996-03-10'),
- (76, 'Harish Rao', 'harish.rao@gmail.com', 'Male', 37, 76, 'Raipur', '1997-04-05'),
- (77, 'Diksha Mehta', 'diksha.mehta@gmail.com', 'Female', 48, 77, 'Faridabad', '1998-05-15'),
- (78, 'Rahul Sharma', 'rahul.sharma@gmail.com', 'Male', 59, 78, 'Bhopal', '1995-06-20'),
- (79, 'Parul Yadav', 'parul.yadav@gmail.com', 'Female', 70, 79, 'Coimbatore', '1999-07-15'),
- (80, 'Ravindra Singh', 'ravindra.singh@gmail.com', 'Male', 81, 80, 'Chennai', '1996-08-01'),
- (81, 'Vishakha Patel', 'vishakha.patel@gmail.com', 'Female', 92, 81, 'Delhi', '1998-09-25'),
- (82, 'Himanshu Joshi', 'himanshu.joshi@gmail.com', 'Male', 3, 82, 'Indore', '1995-10-11'),
- (83, 'Shweta Mehta', 'shweta.mehta@gmail.com', 'Female', 14, 83, 'Lucknow', '1996-11-30'),
- (84, 'Dev Yadav', 'dev.yadav@gmail.com', 'Male', 25, 84, 'Jaipur', '1999-12-05'),
- (85, 'Aditi Singh', 'aditi.singh@gmail.com', 'Female', 36, 85, 'Pune', '1995-01-15'),
- (86, 'Raj Kumar', 'raj.kumar@gmail.com', 'Male', 47, 86, 'Nashik', '1999-02-20'),
- (87, 'Nisha Kaur', 'nisha.kaur@gmail.com', 'Female', 58, 87, 'Ahmedabad', '2000-03-30'),
- (88, 'Sandeep Verma', 'sandeep.verma@gmail.com', 'Male', 69, 88, 'Surat', '1996-04-10'),

-
- (89, 'Kavya Agarwal', 'kavya.agarwal@gmail.com', 'Female', 70, 89, 'Coimbatore', '1998-05-11'),
(90, 'Mohit Iyer', 'mohit.iyer@gmail.com', 'Male', 71, 90, 'Delhi', '1999-06-12'),
(91, 'Vani Sharma', 'vani.sharma@gmail.com', 'Female', 72, 91, 'Mumbai', '1996-07-15'),
(92, 'Nitin Singh', 'nitin.singh@gmail.com', 'Male', 73, 92, 'Bangalore', '1998-08-16'),
(93, 'Meenal Patel', 'meenal.patel@gmail.com', 'Female', 74, 93, 'Chennai', '1999-09-17'),
(94, 'Rajeev Kumar', 'rajeeve.kumar@gmail.com', 'Male', 75, 94, 'Hyderabad', '1996-10-18'),
(95, 'Gurpreet Singh', 'gurpreet.singh@gmail.com', 'Male', 76, 95, 'Kolkata', '1995-11-19'),
(96, 'Rohit Yadav', 'rohit.yadav@gmail.com', 'Male', 77, 96, 'Pune', '1998-12-20'),
(97, 'Bhumika Jain', 'bhumika.jain@gmail.com', 'Female', 78, 97, 'Ahmedabad', '1996-01-21'),
(98, 'Sakshi Singh', 'sakshi.singh@gmail.com', 'Female', 79, 98, 'Coimbatore', '1999-02-22'),
(99, 'Harish Gupta', 'harish.gupta@gmail.com', 'Male', 80, 99, 'Delhi', '1995-03-23'),
(100, 'Priyanka Reddy', 'priyanka.reddy@gmail.com', 'Female', 81, 100, 'Jaipur', '1998-04-24');

9) Employment

```
INSERT INTO Employment (Job_ID, Industry_Type, Salary,  
Start_Date, End_Date) VALUES  
(1, 'Information Technology', 60000.00, '2022-01-01',  
'2025-01-01'),  
(2, 'Healthcare', 50000.00, '2021-02-01', '2024-12-01'),  
(3, 'Finance', 70000.00, '2023-03-01', '2025-03-01'),  
(4, 'Education', 45000.00, '2022-04-01', NULL),  
(5, 'Construction', 55000.00, '2020-05-01', '2023-06-01'),  
(6, 'Manufacturing', 50000.00, '2019-06-01', NULL),  
(7, 'Retail', 40000.00, '2023-07-01', '2025-07-01'),  
(8, 'Hospitality', 35000.00, '2020-08-01', '2023-08-01'),  
(9, 'Transportation', 45000.00, '2022-09-01', NULL),  
(10, 'Telecommunications', 65000.00, '2021-10-01', '2024-10-01'),  
(11, 'Real Estate', 70000.00, '2022-11-01', '2025-11-01'),  
(12, 'Legal', 75000.00, '2020-12-01', NULL),  
(13, 'Marketing', 50000.00, '2023-01-01', '2024-12-01'),  
(14, 'Media', 40000.00, '2021-02-01', NULL),  
(15, 'Agriculture', 30000.00, '2020-03-01', '2023-03-01'),  
(16, 'Information Security', 90000.00, '2023-04-01', '2025-04-01'),  
(17, 'Pharmaceuticals', 80000.00, '2021-05-01', NULL),  
(18, 'Aerospace', 75000.00, '2022-06-01', '2024-06-01'),  
(19, 'Biotechnology', 85000.00, '2023-07-01', '2025-07-01'),  
(20, 'Consulting', 65000.00, '2022-08-01', NULL),  
(21, 'Entertainment', 55000.00, '2023-09-01', NULL),  
(22, 'Insurance', 60000.00, '2021-10-01', '2024-10-01'),  
(23, 'Textiles', 40000.00, '2022-11-01', NULL),  
(24, 'Food Service', 35000.00, '2021-12-01', '2023-12-01'),
```

(25, 'Construction Management', 60000.00, '2023-01-01',
'2025-01-01'),
(26, 'Civil Engineering', 70000.00, '2022-02-01', '2024-02-01'),
(27, 'Environmental Science', 50000.00, '2023-03-01',
'2024-03-01'),
(28, 'Data Analysis', 80000.00, '2023-04-01', NULL),
(29, 'Web Development', 60000.00, '2022-05-01', '2024-05-01'),
(30, 'Graphic Design', 40000.00, '2023-06-01', NULL),
(31, 'User Experience Design', 70000.00, '2021-07-01',
'2024-07-01'),
(32, 'Content Creation', 30000.00, '2022-08-01', NULL),
(33, 'Event Planning', 35000.00, '2020-09-01', NULL),
(34, 'Sales', 45000.00, '2021-10-01', '2024-10-01'),
(35, 'Human Resources', 50000.00, '2023-01-01', NULL),
(36, 'Supply Chain Management', 60000.00, '2021-02-01', NULL),
(37, 'Public Relations', 55000.00, '2022-03-01', '2024-03-01'),
(38, 'Nonprofit Management', 40000.00, '2020-04-01', NULL),
(39, 'Retail Management', 45000.00, '2022-05-01', NULL),
(40, 'Property Management', 50000.00, '2023-06-01', '2024-06-01'),
(41, 'Event Management', 35000.00, '2022-07-01', NULL),
(42, 'Business Development', 60000.00, '2021-08-01',
'2024-08-01'),
(43, 'Research and Development', 70000.00, '2020-09-01', NULL),
(44, 'Project Management', 65000.00, '2023-10-01', '2025-10-01'),
(45, 'Software Development', 80000.00, '2022-11-01', NULL),
(46, 'Data Science', 85000.00, '2023-12-01', NULL),
(47, 'Cybersecurity', 90000.00, '2023-01-01', '2025-01-01'),
(48, 'Cloud Computing', 70000.00, '2021-02-01', NULL),
(49, 'Artificial Intelligence', 95000.00, '2022-03-01', '2024-03-01'),

(50, 'Machine Learning', 80000.00, '2023-04-01', NULL),
(51, 'Blockchain', 85000.00, '2020-05-01', '2024-05-01'),
(52, 'Robotics', 90000.00, '2021-06-01', NULL),
(53, 'Internet of Things', 95000.00, '2023-07-01', '2025-07-01'),
(54, 'Augmented Reality', 70000.00, '2021-08-01', NULL),
(55, 'Virtual Reality', 60000.00, '2020-09-01', '2023-09-01'),
(56, 'Big Data', 80000.00, '2023-10-01', NULL),
(57, 'Digital Marketing', 70000.00, '2021-11-01', '2024-11-01'),
(58, 'E-commerce', 75000.00, '2022-12-01', NULL),
(59, 'SEO', 65000.00, '2023-01-01', NULL),
(60, 'Content Marketing', 60000.00, '2020-02-01', '2023-02-01'),
(61, 'Social Media Management', 55000.00, '2021-03-01', NULL),
(62, 'App Development', 70000.00, '2022-04-01', '2024-04-01'),
(63, 'Quality Assurance', 65000.00, '2023-05-01', NULL),
(64, 'DevOps', 80000.00, '2020-06-01', NULL),
(65, 'Game Development', 75000.00, '2021-07-01', '2024-07-01'),
(66, 'Product Management', 90000.00, '2023-08-01', '2025-08-01'),
(67, 'Technical Support', 50000.00, '2022-09-01', NULL),
(68, 'Sales Management', 65000.00, '2021-10-01', NULL),
(69, 'Business Analysis', 70000.00, '2020-11-01', '2023-11-01'),
(70, 'Financial Analysis', 80000.00, '2023-12-01', NULL),
(71, 'Operations Management', 70000.00, '2021-01-01', NULL),
(72, 'Strategic Planning', 75000.00, '2022-02-01', NULL),
(73, 'Customer Service', 60000.00, '2020-03-01', NULL),
(74, 'Market Research', 55000.00, '2021-04-01', NULL),
(75, 'Policy Analysis', 70000.00, '2023-05-01', NULL),
(76, 'Community Development', 40000.00, '2021-06-01', NULL),
(77, 'Mental Health', 50000.00, '2020-07-01', '2023-07-01'),
(78, 'Consultative Selling', 60000.00, '2022-08-01', '2024-08-01'),

(79, 'Account Management', 55000.00, '2021-09-01', NULL),
(80, 'Risk Management', 70000.00, '2023-10-01', NULL),
(81, 'Facilities Management', 50000.00, '2021-11-01', NULL),
(82, 'Logistics', 60000.00, '2023-12-01', NULL),
(83, 'Energy Management', 75000.00, '2023-01-01', NULL),
(84, 'Health and Safety', 60000.00, '2022-02-01', '2025-02-01'),
(85, 'Environmental Engineering', 80000.00, '2021-03-01', NULL),
(86, 'Sustainability Consulting', 70000.00, '2023-04-01',
'2024-04-01'),
(87, 'Telecommunications Engineering', 85000.00, '2020-05-01',
NULL),
(88, 'Cybersecurity Analysis', 95000.00, '2022-06-01',
'2025-06-01'),
(89, 'Data Governance', 75000.00, '2021-07-01', NULL),
(90, 'Artificial Intelligence Research', 95000.00, '2023-08-01',
NULL),
(91, 'User Interface Design', 70000.00, '2022-09-01', '2024-09-01'),
(92, 'Network Administration', 60000.00, '2021-10-01', NULL),
(93, 'Health Informatics', 70000.00, '2023-11-01', NULL),
(94, 'Pharmaceutical Sales', 75000.00, '2020-12-01', '2023-12-01'),
(95, 'Regulatory Affairs', 80000.00, '2022-01-01', NULL),
(96, 'Nonprofit Fundraising', 40000.00, '2023-02-01',
'2025-02-01'),
(97, 'Training and Development', 60000.00, '2021-03-01', NULL),
(98, 'Corporate Social Responsibility', 65000.00, '2022-04-01',
NULL),
(99, 'International Relations', 70000.00, '2023-05-01', NULL),
(100, 'Business Intelligence', 80000.00, '2022-06-01', NULL);

10) Education Table

```
INSERT INTO Education (Education_ID, Institute_ID, City_ID,
Start_Date, End_Date) VALUES
(1, 1, 15, '2019-08-01', '2022-05-31'),
(2, 2, 34, '2020-01-15', NULL),
(3, 3, 24, '2015-06-20', '2020-05-31'),
(4, 4, 127, '2021-08-01', '2023-06-30'),
(5, 5, 103, '2020-09-01', '2023-12-31'),
(6, 6, 8, '2021-05-15', NULL),
(7, 7, 104, '2018-07-20', '2022-06-30'),
(8, 8, 29, '2019-03-10', '2022-03-10'),
(9, 9, 128, '2021-08-01', NULL),
(10, 10, 105, '2017-11-15', '2020-11-15'),
(11, 11, 106, '2020-02-20', '2022-12-20'),
(12, 12, 107, '2019-04-01', NULL),
(13, 13, 24, '2016-09-01', '2019-09-01'),
(14, 14, 108, '2018-01-01', '2021-01-01'),
(15, 15, 109, '2021-02-01', NULL),
(16, 16, 71, '2020-08-15', NULL),
(17, 17, 34, '2015-05-01', '2020-05-01'),
(18, 18, 34, '2020-08-01', '2022-08-01'),
(19, 19, 34, '2019-09-10', '2021-06-30'),
(20, 20, 21, '2021-10-15', NULL),
(21, 21, 15, '2020-03-20', '2023-03-20'),
(22, 22, 29, '2015-02-01', '2018-06-30'),
(23, 23, 34, '2019-11-11', NULL),
(24, 24, 8, '2021-01-01', '2023-12-01'),
(25, 25, 71, '2018-05-20', '2021-12-31'),
(26, 26, 70, '2017-12-01', NULL),
```

(27, 27, 15, '2019-04-15', '2022-04-15'),
(28, 28, 24, '2016-08-01', NULL),
(29, 29, 110, '2021-03-01', '2024-02-29'),
(30, 30, 2, '2018-07-01', NULL),
(31, 31, 8, '2015-04-10', '2019-04-10'),
(32, 32, 111, '2020-09-01', NULL),
(33, 33, 104, '2017-11-01', '2020-11-01'),
(34, 34, 113, '2019-10-01', '2022-10-01'),
(35, 35, 78, '2020-06-20', NULL),
(36, 36, 109, '2019-01-15', '2023-05-15'),
(37, 37, 29, '2021-12-01', NULL),
(38, 38, 34, '2015-03-20', '2019-03-20'),
(39, 39, 115, '2020-05-01', '2023-04-30'),
(40, 40, 102, '2018-10-01', '2021-10-01'),
(41, 41, 28, '2020-04-15', NULL),
(42, 42, 116, '2016-07-01', '2019-07-01'),
(43, 43, 34, '2021-08-15', NULL),
(44, 44, 34, '2019-12-01', '2022-12-01'),
(45, 45, 86, '2017-02-10', NULL),
(46, 46, 117, '2020-03-15', '2023-09-15'),
(47, 47, 104, '2018-05-01', NULL),
(48, 48, 107, '2016-06-20', '2019-05-20'),
(49, 49, 104, '2020-07-15', NULL),
(50, 50, 24, '2015-09-01', '2019-09-01'),
(51, 51, 110, '2018-01-01', NULL),
(52, 52, 116, '2019-02-20', '2022-02-20'),
(53, 53, 123, '2021-11-11', NULL),
(54, 54, 23, '2016-03-01', NULL),
(55, 55, 29, '2020-01-10', '2023-01-10'),

(56, 56, 25, '2017-08-01', NULL),
(57, 57, 22, '2015-12-01', '2018-12-01'),
(58, 58, 34, '2019-09-20', NULL),
(59, 59, 104, '2020-05-15', NULL),
(60, 60, 8, '2015-10-01', '2019-10-01'),
(61, 61, 34, '2021-02-01', NULL),
(62, 62, 34, '2018-03-10', NULL),
(63, 63, 34, '2019-01-01', '2022-01-01'),
(64, 64, 125, '2017-11-20', NULL),
(65, 65, 126, '2020-04-01', '2023-08-01'),
(66, 66, 129, '2019-07-10', NULL),
(67, 67, 65, '2021-05-01', '2023-05-01'),
(68, 68, 11, '2016-09-01', NULL),
(69, 69, 27, '2015-06-30', '2019-06-30'),
(70, 70, 130, '2018-12-20', NULL),
(71, 71, 25, '2017-02-01', NULL),
(72, 72, 48, '2019-01-15', NULL),
(73, 73, 131, '2020-03-30', NULL),
(74, 74, 8, '2015-08-10', NULL),
(75, 75, 15, '2021-04-20', '2023-03-20'),
(76, 76, 11, '2019-05-15', NULL),
(77, 77, 129, '2020-07-01', NULL),
(78, 78, 133, '2016-10-01', '2019-10-01'),
(79, 79, 78, '2021-08-01', NULL),
(80, 80, 15, '2019-06-15', NULL),
(81, 81, 34, '2015-03-01', '2018-03-01'),
(82, 82, 13, '2020-11-10', NULL),
(83, 83, 123, '2016-07-01', NULL),
(84, 84, 135, '2017-02-20', '2020-02-20'),

```
(85, 85, 50, '2021-09-01', NULL),
(86, 86, 43, '2019-08-20', NULL),
(87, 87, 27, '2016-04-10', '2020-04-10'),
(88, 88, 136, '2020-01-01', NULL),
(89, 89, 1, '2018-12-31', '2022-12-31'),
(90, 90, 20, '2019-05-15', NULL),
(91, 91, 137, '2021-07-20', NULL),
(92, 92, 4, '2016-03-01', '2019-03-01'),
(93, 93, 138, '2018-11-10', NULL),
(94, 94, 123, '2019-09-15', NULL),
(95, 95, 140, '2020-01-01', NULL),
(96, 96, 56, '2015-06-01', NULL),
(97, 97, 21, '2021-03-10', NULL),
(98, 98, 141, '2016-08-01', NULL),
(99, 99, 25, '2019-10-20', NULL),
(100, 100, 14, '2017-12-01', NULL);
```

11) Migration_Event Table

```
INSERT INTO Migration_Event (Migration_ID, Youngster_ID,
Duration, From_Region, To_Region, Migration_Date)
VALUES
(1, 1, 5, 5, 10, '2018-03-15'),
(2, 2, 12, 20, 15, '2019-06-25'),
(3, 3, 18, 25, 30, '2021-11-05'),
(4, 1, 9, 35, 40, '2020-08-10'),
(5, 5, 6, 10, 45, '2015-09-12'),
(6, 6, 14, 50, 55, '2017-01-20'),
(7, 7, 11, 60, 65, '2022-02-28'),
```

(8, 8, 7, 70, 75, '2020-12-01'),
(9, 9, 15, 80, 85, '2016-04-17'),
(10, 10, 19, 90, 95, '2023-07-22'),
(11, 1, 10, 100, 105, '2014-10-14'),
(12, 12, 17, 110, 115, '2019-05-19'),
(13, 13, 13, 120, 125, '2022-09-30'),
(14, 14, 8, 130, 135, '2021-03-11'),
(15, 15, 16, 140, 1, '2015-06-09'),
(16, 16, 4, 2, 6, '2017-02-05'),
(17, 17, 5, 3, 9, '2024-01-01'),
(18, 18, 7, 4, 14, '2020-11-11'),
(19, 19, 9, 5, 16, '2018-03-22'),
(20, 20, 18, 6, 18, '2019-07-15'),
(21, 21, 3, 7, 20, '2022-12-12'),
(22, 22, 10, 8, 21, '2016-09-25'),
(23, 23, 15, 9, 23, '2014-04-30'),
(24, 24, 12, 10, 24, '2023-05-05'),
(25, 25, 8, 11, 26, '2022-08-18'),
(26, 26, 11, 12, 28, '2015-11-09'),
(27, 27, 2, 13, 30, '2018-10-14'),
(28, 28, 6, 14, 32, '2019-02-20'),
(29, 29, 14, 15, 34, '2021-01-05'),
(30, 30, 5, 16, 36, '2020-03-10'),
(31, 31, 17, 17, 38, '2016-06-21'),
(32, 32, 19, 18, 40, '2022-07-01'),
(33, 33, 3, 19, 41, '2015-08-28'),
(34, 34, 8, 20, 42, '2024-03-15'),
(35, 35, 7, 21, 44, '2018-01-10'),
(36, 36, 10, 22, 46, '2017-04-12'),

(37, 37, 12, 23, 48, '2023-10-30'),
(38, 38, 6, 24, 50, '2020-11-15'),
(39, 39, 14, 25, 52, '2019-12-22'),
(40, 40, 11, 26, 54, '2015-03-17'),
(41, 41, 8, 27, 56, '2021-05-08'),
(42, 42, 10, 28, 58, '2022-02-18'),
(43, 43, 9, 29, 60, '2014-10-12'),
(44, 44, 7, 30, 62, '2023-04-05'),
(45, 45, 6, 31, 63, '2017-08-15'),
(46, 46, 4, 32, 64, '2018-01-25'),
(47, 47, 2, 33, 66, '2022-06-09'),
(48, 48, 10, 34, 67, '2016-09-13'),
(49, 49, 5, 35, 68, '2015-02-20'),
(50, 50, 11, 36, 69, '2019-10-18'),
(51, 51, 7, 37, 70, '2021-03-14'),
(52, 52, 9, 38, 72, '2020-11-05'),
(53, 53, 13, 39, 74, '2019-09-12'),
(54, 54, 14, 40, 75, '2016-03-22'),
(55, 55, 8, 41, 76, '2018-09-11'),
(56, 56, 11, 42, 78, '2019-11-20'),
(57, 57, 15, 43, 80, '2023-05-30'),
(58, 58, 10, 44, 81, '2017-06-12'),
(59, 59, 12, 45, 83, '2020-12-08'),
(60, 60, 6, 46, 84, '2021-01-29'),
(61, 61, 4, 47, 85, '2019-04-18'),
(62, 62, 13, 48, 86, '2015-10-22'),
(63, 63, 14, 49, 87, '2022-07-01'),
(64, 64, 5, 50, 88, '2016-02-14'),
(65, 65, 8, 51, 89, '2023-03-15'),

(66, 66, 6, 52, 90, '2019-09-12'),
(67, 67, 11, 53, 91, '2015-08-25'),
(68, 68, 18, 54, 92, '2020-11-05'),
(69, 69, 9, 55, 93, '2021-04-18'),
(70, 70, 15, 56, 94, '2017-01-29'),
(71, 71, 3, 57, 95, '2023-01-15'),
(72, 72, 2, 58, 96, '2018-03-01'),
(73, 73, 4, 59, 97, '2015-12-17'),
(74, 74, 10, 60, 98, '2019-05-22'),
(75, 75, 12, 61, 99, '2020-07-28'),
(76, 76, 8, 62, 100, '2014-09-15'),
(77, 77, 6, 63, 101, '2022-11-30'),
(78, 78, 14, 64, 102, '2021-02-25'),
(79, 79, 11, 65, 103, '2018-04-30'),
(80, 80, 7, 66, 104, '2016-08-02'),
(81, 81, 5, 67, 105, '2014-12-21'),
(82, 82, 10, 68, 106, '2021-07-30'),
(83, 83, 12, 69, 107, '2017-05-09'),
(84, 84, 9, 70, 108, '2019-09-23'),
(85, 85, 3, 71, 109, '2022-02-10'),
(86, 86, 14, 72, 110, '2015-01-15'),
(87, 87, 6, 73, 111, '2021-03-30'),
(88, 88, 13, 74, 112, '2020-05-17'),
(89, 89, 11, 75, 113, '2016-06-19'),
(90, 90, 10, 76, 114, '2019-07-01'),
(91, 91, 17, 77, 115, '2023-08-05'),
(92, 92, 4, 78, 116, '2021-04-18'),
(93, 93, 14, 79, 117, '2017-10-20'),
(94, 94, 15, 80, 118, '2018-12-15'),

(95, 95, 6, 81, 119, '2020-02-01'),
(96, 96, 12, 82, 120, '2015-11-17'),
(97, 97, 8, 83, 121, '2022-09-30'),
(98, 98, 9, 84, 122, '2023-01-10'),
(99, 99, 11, 85, 123, '2019-05-12'),
(100, 100, 3, 86, 124, '2020-11-01'),
(101, 1, 19, 87, 125, '2022-03-20'),
(102, 2, 17, 88, 126, '2020-10-15'),
(103, 3, 10, 89, 127, '2018-06-25'),
(104, 4, 5, 90, 128, '2016-08-14'),
(105, 5, 8, 91, 129, '2022-11-19'),
(106, 6, 12, 92, 130, '2021-07-09'),
(107, 7, 10, 93, 131, '2014-04-10'),
(108, 8, 9, 94, 132, '2019-02-23'),
(109, 9, 6, 95, 133, '2018-12-09'),
(110, 10, 4, 96, 134, '2021-11-20'),
(111, 11, 2, 97, 135, '2017-03-25'),
(112, 12, 16, 98, 136, '2020-12-03'),
(113, 13, 13, 99, 137, '2023-06-30'),
(114, 14, 11, 100, 138, '2018-05-15'),
(115, 15, 19, 101, 139, '2019-07-11'),
(116, 16, 5, 102, 140, '2022-03-04'),
(117, 17, 10, 103, 1, '2021-10-19'),
(118, 18, 6, 104, 2, '2016-08-28'),
(119, 19, 3, 105, 3, '2022-12-22'),
(120, 20, 9, 106, 4, '2023-05-14'),
(121, 21, 8, 107, 5, '2015-11-29'),
(122, 22, 17, 108, 6, '2019-03-16'),
(123, 23, 4, 109, 7, '2018-02-19'),

(124, 24, 12, 110, 8, '2022-08-12'),
(125, 25, 6, 111, 9, '2017-09-25'),
(126, 26, 18, 112, 10, '2019-04-05'),
(127, 27, 5, 113, 11, '2020-07-23'),
(128, 28, 11, 114, 12, '2016-10-15'),
(129, 29, 9, 115, 13, '2021-01-18'),
(130, 30, 3, 116, 14, '2023-06-09'),
(131, 31, 4, 117, 15, '2018-08-17'),
(132, 32, 7, 118, 16, '2020-01-10'),
(133, 33, 6, 119, 17, '2015-11-30'),
(134, 34, 13, 120, 18, '2022-10-14'),
(135, 35, 17, 121, 19, '2019-12-01'),
(136, 36, 15, 122, 20, '2023-03-11'),
(137, 37, 14, 123, 21, '2020-10-07'),
(138, 38, 12, 124, 22, '2016-04-20'),
(139, 39, 8, 125, 23, '2021-05-30'),
(140, 40, 5, 126, 24, '2015-07-15'),
(141, 41, 9, 127, 25, '2018-09-19'),
(142, 42, 11, 128, 26, '2022-01-25'),
(143, 43, 10, 129, 27, '2019-10-15'),
(144, 44, 8, 130, 28, '2016-11-08'),
(145, 45, 6, 131, 29, '2021-06-10'),
(146, 46, 7, 132, 30, '2020-12-12'),
(147, 47, 14, 133, 31, '2022-08-21'),
(148, 48, 3, 134, 32, '2015-04-01'),
(149, 49, 2, 135, 33, '2019-05-17'),
(150, 50, 1, 136, 34, '2023-02-02');

12) Opportunities Table

INSERT INTO Opportunities (Opportunity_Type, Location, Employer_ID, Salary_Benefits, Eligibility_Criteria) VALUES
('Internship', 'Mumbai', 56, '₹15,000 per month', 'Undergraduate'),
('Full-time', 'Delhi', 23, '₹60,000 per month', 'Postgraduate'),
('Part-time', 'Bangalore', 89, '₹25,000 per month', 'Undergraduate'),
('Contract', 'Hyderabad', 45, '₹50,000 per month', 'Any Graduate'),
('Freelance', 'Chennai', 78, 'Project-based', 'Experience preferred'),
('Internship', 'Kolkata', 34, '₹12,000 per month', 'Undergraduate'),
('Full-time', 'Pune', 91, '₹70,000 per month', 'Postgraduate'),
('Part-time', 'Ahmedabad', 10, '₹20,000 per month', 'Diploma'),
('Contract', 'Jaipur', 82, '₹55,000 per month', 'Postgraduate'),
('Freelance', 'Lucknow', 37, 'Project-based', 'Experience preferred'),
('Internship', 'Surat', 66, '₹18,000 per month', 'Undergraduate'),
('Full-time', 'Nagpur', 53, '₹62,000 per month', 'Postgraduate'),
('Part-time', 'Coimbatore', 99, '₹22,000 per month', 'Any Graduate'),
('Contract', 'Visakhapatnam', 18, '₹48,000 per month', 'Experience preferred'),
('Freelance', 'Vijayawada', 84, 'Project-based', 'Undergraduate'),
('Internship', 'Indore', 30, '₹14,000 per month', 'Undergraduate'),
('Full-time', 'Mysore', 72, '₹68,000 per month', 'Postgraduate'),
('Part-time', 'Nashik', 4, '₹21,000 per month', 'Any Graduate'),
('Contract', 'Aurangabad', 60, '₹53,000 per month', 'Postgraduate'),
('Freelance', 'Rajkot', 26, 'Project-based', 'Experience preferred'),
('Internship', 'Vadodara', 77, '₹16,000 per month', 'Undergraduate'),
('Full-time', 'Patna', 11, '₹64,000 per month', 'Postgraduate'),
('Part-time', 'Bhopal', 49, '₹24,000 per month', 'Any Graduate'),

('Contract', 'Gwalior', 67, '₹57,000 per month', 'Experience preferred'),
('Freelance', 'Agra', 93, 'Project-based', 'Undergraduate'),
('Internship', 'Dehradun', 32, '₹15,000 per month', 'Undergraduate'),
('Full-time', 'Ranchi', 90, '₹69,000 per month', 'Postgraduate'),
('Part-time', 'Srinagar', 2, '₹23,000 per month', 'Any Graduate'),
('Contract', 'Jammu', 54, '₹52,000 per month', 'Postgraduate'),
('Freelance', 'Shimla', 71, 'Project-based', 'Experience preferred'),
('Internship', 'Gangtok', 16, '₹17,000 per month', 'Undergraduate'),
('Full-time', 'Imphal', 14, '₹65,000 per month', 'Postgraduate'),
('Part-time', 'Aizawl', 88, '₹20,000 per month', 'Any Graduate'),
('Contract', 'Kohima', 40, '₹49,000 per month', 'Experience preferred'),
('Freelance', 'Itanagar', 41, 'Project-based', 'Undergraduate'),
('Internship', 'Agartala', 86, '₹13,000 per month', 'Undergraduate'),
('Full-time', 'Dibrugarh', 29, '₹63,000 per month', 'Postgraduate'),
('Part-time', 'Guwahati', 1, '₹22,000 per month', 'Any Graduate'),
('Contract', 'Tawang', 33, '₹50,000 per month', 'Postgraduate'),
('Freelance', 'Silchar', 5, 'Project-based', 'Experience preferred'),
('Internship', 'Tezpur', 70, '₹19,000 per month', 'Undergraduate'),
('Full-time', 'Tirupati', 87, '₹66,000 per month', 'Postgraduate'),
('Part-time', 'Kakinada', 75, '₹25,000 per month', 'Any Graduate'),
('Contract', 'Vellore', 57, '₹56,000 per month', 'Postgraduate'),
('Freelance', 'Tirunelveli', 46, 'Project-based', 'Experience preferred'),
('Internship', 'Thiruvananthapuram', 39, '₹18,000 per month', 'Undergraduate'),
('Full-time', 'Nellore', 65, '₹61,000 per month', 'Postgraduate'),
('Part-time', 'Tiruvallur', 38, '₹22,500 per month', 'Any Graduate'),

('Contract', 'Chennai', 48, '₹58,000 per month', 'Postgraduate'),
('Freelance', 'Madurai', 79, 'Project-based', 'Experience preferred'),
('Internship', 'Puducherry', 42, '₹16,500 per month',
'Undergraduate'),
('Full-time', 'Kozhikode', 12, '₹67,000 per month', 'Postgraduate'),
('Part-time', 'Kottayam', 3, '₹24,500 per month', 'Any Graduate'),
('Contract', 'Kannur', 74, '₹52,500 per month', 'Postgraduate'),
('Freelance', 'Malappuram', 97, 'Project-based', 'Experience preferred'),
('Internship', 'Ernakulam', 73, '₹15,500 per month',
'Undergraduate'),
('Full-time', 'Palakkad', 8, '₹64,500 per month', 'Postgraduate'),
('Part-time', 'Idukki', 15, '₹21,500 per month', 'Any Graduate'),
('Contract', 'Pathanamthitta', 50, '₹54,500 per month',
'Postgraduate'),
('Freelance', 'Thiruvananthapuram', 44, 'Project-based', 'Experience preferred'),
('Internship', 'Trivandrum', 35, '₹18,500 per month',
'Undergraduate'),
('Full-time', 'Thiruvananthapuram', 20, '₹66,500 per month',
'Postgraduate'),
('Part-time', 'Chengannur', 80, '₹23,500 per month', 'Any Graduate'),
('Contract', 'Muvattupuzha', 68, '₹58,500 per month',
'Postgraduate'),
('Freelance', 'Kottayam', 27, 'Project-based', 'Experience preferred'),
('Internship', 'Alappuzha', 24, '₹15,800 per month',
'Undergraduate'),

('Full-time', 'Kollam', 61, '₹65,800 per month', 'Postgraduate'),
('Part-time', 'Kottayam', 83, '₹22,800 per month', 'Any Graduate'),
('Contract', 'Thodupuzha', 92, '₹52,800 per month', 'Postgraduate'),
('Freelance', 'Punalur', 6, 'Project-based', 'Experience preferred'),
('Internship', 'Neyyattinkara', 69, '₹17,200 per month',
'Undergraduate'),
('Full-time', 'Munnar', 99, '₹63,800 per month', 'Postgraduate'),
('Part-time', 'Varkala', 71, '₹24,200 per month', 'Any Graduate'),
('Contract', 'Kumarakom', 88, '₹57,200 per month', 'Postgraduate'),
('Freelance', 'Adoor', 25, 'Project-based', 'Experience preferred');

13) Enrolled

```
INSERT INTO Enrolled (Youngster_ID, Institute_ID,  
Enrollment_Date, Completion_Status, Credits, Grade) VALUES  
(1, 36, '2020-08-15', TRUE, 30, 'A'),  
(2, 8, '2019-09-10', FALSE, 28, 'B'),  
(3, 43, '2021-01-12', TRUE, 34, 'A'),  
(4, 10, '2022-07-20', TRUE, 40, 'A'),  
(5, 19, '2018-05-25', FALSE, 20, 'C'),  
(6, 64, '2020-11-30', TRUE, 32, 'B'),  
(7, 52, '2021-02-14', TRUE, 36, 'A'),  
(8, 27, '2022-06-18', FALSE, 22, 'C'),  
(9, 49, '2019-10-11', TRUE, 30, 'B'),  
(10, 15, '2023-01-01', TRUE, 28, 'A'),  
(11, 60, '2020-03-30', FALSE, 25, 'B'),  
(12, 30, '2021-04-12', TRUE, 35, 'A'),  
(13, 78, '2019-08-21', TRUE, 31, 'A'),
```

(14, 42, '2022-02-16', FALSE, 27, 'C'),
(15, 54, '2020-12-01', TRUE, 29, 'B'),
(16, 70, '2021-03-05', TRUE, 38, 'A'),
(17, 33, '2018-06-29', FALSE, 21, 'C'),
(18, 5, '2019-11-14', TRUE, 36, 'B'),
(19, 66, '2021-08-20', TRUE, 34, 'A'),
(20, 88, '2020-04-25', TRUE, 37, 'A'),
(21, 81, '2019-10-30', FALSE, 29, 'B'),
(22, 94, '2022-07-09', TRUE, 40, 'A'),
(23, 37, '2018-05-18', TRUE, 31, 'B'),
(24, 91, '2020-11-22', FALSE, 30, 'C'),
(25, 47, '2021-02-28', TRUE, 35, 'A'),
(26, 73, '2019-09-15', TRUE, 33, 'B'),
(27, 50, '2021-01-01', FALSE, 26, 'C'),
(28, 14, '2020-05-05', TRUE, 30, 'A'),
(29, 63, '2019-08-16', TRUE, 32, 'B'),
(30, 12, '2021-03-18', FALSE, 27, 'C'),
(31, 59, '2019-02-22', TRUE, 35, 'A'),
(32, 84, '2021-06-17', TRUE, 38, 'A'),
(33, 39, '2020-07-07', FALSE, 24, 'B'),
(34, 11, '2019-03-13', TRUE, 28, 'A'),
(35, 92, '2021-01-30', TRUE, 36, 'B'),
(36, 76, '2020-08-02', TRUE, 40, 'A'),
(37, 68, '2019-10-15', FALSE, 22, 'C'),
(38, 45, '2021-05-05', TRUE, 29, 'A'),
(39, 86, '2020-04-12', TRUE, 31, 'B'),
(40, 80, '2019-09-21', TRUE, 35, 'A'),
(41, 26, '2021-03-02', FALSE, 24, 'C'),
(42, 99, '2020-12-29', TRUE, 36, 'B'),

(43, 1, '2022-11-05', TRUE, 40, 'A'),
(44, 74, '2019-07-04', FALSE, 21, 'C'),
(45, 24, '2021-08-17', TRUE, 33, 'A'),
(46, 67, '2020-09-28', TRUE, 30, 'B'),
(47, 95, '2019-05-15', FALSE, 27, 'C'),
(48, 62, '2021-02-20', TRUE, 38, 'A'),
(49, 18, '2020-03-06', TRUE, 34, 'B'),
(50, 53, '2021-07-01', TRUE, 30, 'A'),
(51, 34, '2019-04-14', TRUE, 29, 'B'),
(52, 35, '2022-05-10', FALSE, 20, 'C'),
(53, 8, '2019-11-21', TRUE, 37, 'A'),
(54, 90, '2020-06-15', TRUE, 32, 'B'),
(55, 56, '2021-01-05', TRUE, 40, 'A'),
(56, 82, '2022-04-11', FALSE, 24, 'C'),
(57, 13, '2020-08-13', TRUE, 30, 'A'),
(58, 44, '2021-03-03', TRUE, 33, 'B'),
(59, 72, '2019-06-19', TRUE, 35, 'A'),
(60, 9, '2022-07-08', FALSE, 22, 'C'),
(61, 48, '2020-12-12', TRUE, 29, 'B'),
(62, 58, '2021-10-17', TRUE, 40, 'A'),
(63, 65, '2019-07-24', FALSE, 30, 'C'),
(64, 97, '2022-11-30', TRUE, 28, 'B'),
(65, 77, '2020-01-01', TRUE, 36, 'A'),
(66, 3, '2021-02-22', FALSE, 22, 'C'),
(67, 31, '2019-05-30', TRUE, 37, 'B'),
(68, 20, '2020-08-26', TRUE, 38, 'A'),
(69, 41, '2021-07-07', FALSE, 26, 'C'),
(70, 16, '2022-10-10', TRUE, 35, 'B'),
(71, 79, '2019-03-29', TRUE, 34, 'A'),

(72, 88, '2020-02-10', TRUE, 30, 'B'),
(73, 12, '2021-04-14', TRUE, 28, 'A'),
(74, 25, '2019-06-05', FALSE, 23, 'C'),
(75, 69, '2021-09-17', TRUE, 29, 'A'),
(76, 83, '2019-11-04', TRUE, 35, 'B'),
(77, 96, '2020-10-31', FALSE, 22, 'C'),
(78, 55, '2021-05-16', TRUE, 40, 'A'),
(79, 61, '2019-07-20', TRUE, 31, 'B'),
(80, 38, '2020-06-28', TRUE, 36, 'A'),
(81, 14, '2021-02-04', FALSE, 24, 'C'),
(82, 75, '2020-08-18', TRUE, 30, 'B'),
(83, 7, '2019-05-15', TRUE, 34, 'A'),
(84, 40, '2022-01-12', FALSE, 23, 'C'),
(85, 92, '2020-03-30', TRUE, 29, 'A'),
(86, 45, '2019-08-23', TRUE, 32, 'B'),
(87, 99, '2021-11-05', TRUE, 31, 'A'),
(88, 2, '2020-09-12', TRUE, 36, 'B'),
(89, 4, '2022-06-10', FALSE, 27, 'C'),
(90, 87, '2021-01-20', TRUE, 40, 'A'),
(91, 6, '2019-11-16', TRUE, 28, 'B'),
(92, 15, '2020-04-19', FALSE, 25, 'C'),
(93, 93, '2021-03-28', TRUE, 39, 'A'),
(94, 1, '2022-05-01', TRUE, 35, 'B'),
(95, 89, '2019-10-14', TRUE, 32, 'A'),
(96, 17, '2020-02-09', TRUE, 28, 'B'),
(97, 100, '2021-07-11', FALSE, 22, 'C'),
(98, 39, '2022-03-21', TRUE, 37, 'A'),
(99, 86, '2020-11-09', TRUE, 30, 'B'),
(100, 49, '2021-06-04', FALSE, 20, 'C);

14) Youngster_Phone Table

```
INSERT INTO Youngster_Phone (Youngster_ID, Phone_no)  
VALUES  
(1, '9876543210'),  
(1, '9123456789'),  
(2, '9876543211'),  
(3, '9876543212'),  
(3, '9123456790'),  
(4, '9876543213'),  
(5, '9876543214'),  
(6, '9876543215'),  
(7, '9876543216'),  
(8, '9876543217'),  
(9, '9876543218'),  
(10, '9876543219'),  
(11, '9876543220'),  
(12, '9876543221'),  
(13, '9876543222'),  
(14, '9876543223'),  
(15, '9876543224'),  
(16, '9876543225'),  
(17, '9876543226'),  
(18, '9876543227'),  
(19, '9876543228'),  
(20, '9876543229'),  
(21, '9876543230'),  
(22, '9876543231'),  
(23, '9876543232'),  
(24, '9876543233'),
```

(25, '9876543234'),
(26, '9876543235'),
(27, '9876543236'),
(28, '9876543237'),
(29, '9876543238'),
(30, '9876543239'),
(31, '9876543240'),
(32, '9876543241'),
(33, '9876543242'),
(34, '9876543243'),
(35, '9876543244'),
(36, '9876543245'),
(37, '9876543246'),
(38, '9876543247'),
(39, '9876543248'),
(40, '9876543249'),
(41, '9876543250'),
(42, '9876543251'),
(43, '9876543252'),
(44, '9876543253'),
(45, '9876543254'),
(46, '9876543255'),
(47, '9876543256'),
(48, '9876543257'),
(49, '9876543258'),
(50, '9876543259'),
(51, '9876543260'),
(52, '9876543261'),
(53, '9876543262'),

(54, '9876543263'),
(55, '9876543264'),
(56, '9876543265'),
(57, '9876543266'),
(58, '9876543267'),
(59, '9876543268'),
(60, '9876543269'),
(61, '9876543270'),
(62, '9876543271'),
(63, '9876543272'),
(64, '9876543273'),
(65, '9876543274'),
(66, '9876543275'),
(67, '9876543276'),
(68, '9876543277'),
(69, '9876543278'),
(70, '9876543279'),
(71, '9876543280'),
(72, '9876543281'),
(73, '9876543282'),
(74, '9876543283'),
(75, '9876543284'),
(76, '9876543285'),
(77, '9876543286'),
(78, '9876543287'),
(79, '9876543288'),
(80, '9876543289'),
(81, '9876543290'),
(82, '9876543291'),

(83, '9876543292'),
(84, '9876543293'),
(85, '9876543294'),
(86, '9876543295'),
(87, '9876543296'),
(88, '9876543297'),
(89, '9876543298'),
(90, '9876543299'),
(91, '9876543300'),
(92, '9876543301'),
(93, '9876543302'),
(94, '9876543303'),
(95, '9876543304'),
(96, '9876543305'),
(97, '9876543306'),
(98, '9876543307'),
(99, '9876543308'),
(100, '9876543309');

State Table

The screenshot shows the pgAdmin 4 interface with the 'Object Explorer' on the left and the 'Query Editor' on the right. The 'Object Explorer' displays the database structure, including servers, databases, and tables. The 'Query Editor' shows a query result for the 'state' table.

Query Result:

state_id	state_name
1	Andhra Pradesh
2	Arunachal Pradesh
3	Assam
4	Bihar
5	Chhattisgarh
6	Goa
7	Gujarat
8	Haryana
9	Himachal Pradesh
10	Jharkhand
11	Karnataka
12	Kerala
13	Madhya Pradesh
14	Maharashtra
15	Manipur
16	Meghalaya
17	Mizoram
18	Nagaland
19	Odisha

Total rows: 36 of 36 Query complete 00:00:00.258

City Table

The screenshot shows the pgAdmin 4 interface with the 'Object Explorer' on the left and the 'Query Editor' on the right. The 'Object Explorer' displays the database structure, including servers, databases, and tables. The 'Query Editor' shows a query result for the 'city' table.

Query Result:

city_id	city_name	state_id
1	Vizianapatnam	1
2	Vijayawada	1
3	Itanagar	2
4	Guwahati	3
5	Patna	4
6	Raipur	5
7	Panaji	6
8	Ahmedabad	7
9	Gurugram	8
10	Shimla	9
11	Ranchi	10
12	Bengaluru	11
13	Thiruvananthapuram	12
14	Bhopal	13
15	Mumbai	14
16	Imphal	15
17	Shillong	16
18	Aizawl	17
19	Kohima	18

Total rows: 141 of 141 Query complete 00:00:00.081

Institute table

The screenshot shows the pgAdmin 4 interface with the 'Institute' table selected in the Object Explorer. The table contains 18 rows of data about various institutes.

	institution_id	name	tuition_fees	address	website	accreditation_status
1	6	IIM Ahmedabad	250000.00	Ahmedabad, Gujarat	http://www.iima.ac.in	true
2	86	Kakatiya University	50000.00	Warangal, Telangana	http://www.katkiya.ac.in	true
3	72	Karnataka State Open University	40000.00	Mysuru, Karnataka	http://www.ksoou.ac.karnataka.gov.in	true
4	85	Kochi University of Science and Technology	70000.00	Kochi, Kerala	http://www.kust.edu.in	true
5	96	Guru Nanak Dev Engineering College	70000.00	Ludhiana, Punjab	http://www.gndec.ac.in	true
6	67	Indian Institute of Technology, Dhanbad	200000.00	Dhanbad, Jharkhand	http://www.iiitm.ac.in	true
7	26	Banaras Hindu University	50000.00	Varanasi, Uttar Pradesh	http://www.bhu.ac.in	true
8	16	Shiv Nadar University	180000.00	Greater Noida, Uttar Pradesh	http://www.snu.edu.in	true
9	25	Amity University	100000.00	Noida, Uttar Pradesh	http://www.amity.edu	true
10	35	Shivaji University	40000.00	Kolhapur, Maharashtra	http://www.unishvaji.ac.in	true
11	79	Shivaji University	40000.00	Kolhapur, Maharashtra	http://www.unishvaji.ac.in	true
12	45	Mahatma Gandhi University	50000.00	Kottayam, Kerala	http://www.mgu.ac.in	true
13	5	IIT Kharagpur	200000.00	Kharagpur, West Bengal	http://www.iitkgp.ac.in	true
14	7	IIM Bangalore	250000.00	Bangalore, Karnataka	http://www.iimb.ac.in	true
15	33	IIT Bangalore	120000.00	Bangalore, Karnataka	http://www.iitb.ac.in	true
16	47	REVA University	60000.00	Bangalore, Karnataka	http://www.reva.edu.in	true
17	49	M S Ramaiah University	80000.00	Bangalore, Karnataka	http://www.msrus.ac.in	true
18	59	National Law School of India University	60000.00	Bangalore, Karnataka	http://www.nls.ac.in	true

Employer table

The screenshot shows the pgAdmin 4 interface with the 'Employer' table selected in the Object Explorer. The table contains 19 rows of data about various employers.

	employer_id	employer_name	location	salary_benefits
1	1	Tata Consultancy Services	Mumbai	800000
2	2	Infosys	Bengaluru	750000
3	3	Wipro	Bengaluru	700000
4	4	HCL Technologies	Noida	720000
5	5	Accenture	Gurgaon	900000
6	6	Cognizant	Chennai	680000
7	7	Tech Mahindra	Pune	640000
8	8	Capgemini	Mumbai	750000
9	9	IBM	Bengaluru	850000
10	10	Oracle	Hyderabad	900000
11	11	SAP	Gurgaon	950000
12	12	Dell	Bengaluru	780000
13	13	Intel	Hyderabad	830000
14	14	Microsoft	Hyderabad	1000000
15	15	Amazon	Bengaluru	1200000
16	16	Google	Hyderabad	1100000
17	17	Facebook	Hyderabad	1150000
18	18	Adobe	Noida	900000
19	19	Salesforce	Bengaluru	950000

Health_Facility table

The screenshot shows the pgAdmin 4 interface with the 'Health_Facility' table selected in the Object Explorer. The table has four columns: facility_id, name, facility_type, and bed_capacity. The data consists of 19 rows of hospital information.

facility_id	name	facility_type	bed_capacity
1	All India Institute of Medical Sciences	Hospital	800
2	Post Graduate Institute of Medical Education and Research	Hospital	1500
3	Tata Memorial Hospital	Cancer Hospital	600
4	Fortis Hospital	Multi-Specialty Hospital	200
5	Apollo Hospitals	Multi-Specialty Hospital	400
6	Max Super Speciality Hospital	Multi-Specialty Hospital	300
7	Narayana Health	Multi-Specialty Hospital	1000
8	Medanta – The Medicity	Multi-Specialty Hospital	800
9	Kokilaben Dhirubhai Ambani Hospital	Multi-Specialty Hospital	300
10	Manipal Hospital	Multi-Specialty Hospital	500
11	Lilavati Hospital	Multi-Specialty Hospital	350
12	Sri Ramachandra Medical Centre	Hospital	700
13	Sankara Nethravaya	Eye Hospital	250
14	P.D. Hinduja Hospital	Multi-Specialty Hospital	300
15	Care Hospital	Multi-Specialty Hospital	350
16	Jaypee Hospital	Multi-Specialty Hospital	200
17	Sitaram Bhartia Institute of Science and Research	Hospital	100
18	HCG Cancer Centre	Cancer Hospital	150
19	Aster CMI Hospital	Multi-Specialty Hospital	250

Climate table

The screenshot shows the pgAdmin 4 interface with the 'Climate' table selected in the Object Explorer. The table has four columns: city_id, policy_id, average_temp, and climate_type. The data consists of 19 rows of climate information.

city_id	policy_id	average_temp	climate_type
1	1	25.00	Tropical
2	1	26.50	Tropical
3	1	24.00	Tropical
4	2	30.00	Tropical
5	2	31.00	Tropical
6	2	28.50	Tropical
7	3	14.00	Temperate
8	3	15.00	Temperate
9	3	13.50	Temperate
10	4	20.00	Temperate
11	4	21.00	Temperate
12	4	19.50	Temperate
13	5	8.00	Cold
14	5	9.00	Cold
15	5	7.50	Cold
16	6	27.00	Tropical
17	6	26.50	Tropical
18	6	28.50	Tropical
19	7	15.00	Mediterranean

Government_Policy table

The screenshot shows the pgAdmin 4 interface with the following details:

- Object Explorer:** Shows the database structure under "Youngsters_db".
- Dashboard:** Shows the query `select * from government_policy;`.
- Data Output:** Displays the results of the query in a table format.
- Table Structure:**

policy_id	policy_name	policy_type	state_id
1	Migration Support Scheme	Migration	1
2	Interstate Migrant Workers Act	Migration	3
3	Bihar Migrant Labour Welfare Program	Migration	4
4	Chhattisgarh Migrant Assistance	Migration	5
5	Goa Migrant Worker Act	Migration	6
6	Gujarat Migrant Welfare Board	Migration	7
7	Haryana Migrant Worker Program	Migration	8
8	Himachal Pradesh Migrant Assistance	Migration	9
9	Jharkhand Migrant Support	Migration	10
10	Karnataka Migrant Worker Policy	Migration	11
11	Kerala Migrant Welfare Program	Migration	12
12	Madhya Pradesh Migration Assistance	Migration	13
13	Maharashtra Migrant Worker Act	Migration	14
14	Odisha Migrant Labour Welfare Scheme	Migration	19
15	Punjab Migrant Support Program	Migration	20
16	Rajasthan Migrant Worker Program	Migration	21
17	Tamil Nadu Migrant Assistance	Migration	23
18	Telangana Migrant Welfare Act	Migration	24
19	West Bengal Migrant Assistance	Migration	28
- Message Bar:** Shows "Total rows: 100 of 100 Query complete 00:00:00.082".
- System Bar:** Shows system status including "ENG IN", "1:34 PM", and "11/5/2024".

Employment table

The screenshot shows the pgAdmin 4 interface with the following details:

- Object Explorer:** Shows the database structure under "Youngsters_db".
- Dashboard:** Shows the query `select * from Employment;`.
- Data Output:** Displays the results of the query in a table format.
- Table Structure:**

job_id	industry_type	salary	start_date	end_date
1	Information Technology	60000.00	2022-01-01	2025-01-01
2	Healthcare	50000.00	2021-02-01	2024-12-01
3	Finance	70000.00	2023-03-01	2025-09-01
4	Education	45000.00	2022-04-01	[null]
5	Construction	55000.00	2020-05-01	2023-06-01
6	Manufacturing	50000.00	2019-06-01	[null]
7	Retail	40000.00	2023-07-01	2025-07-01
8	Hospitality	35000.00	2020-08-01	2023-08-01
9	Transportation	45000.00	2022-09-01	[null]
10	Telecommunications	65000.00	2021-10-01	2024-10-01
11	Real Estate	70000.00	2022-11-01	2025-11-01
12	Legal	75000.00	2020-12-01	[null]
13	Marketing	50000.00	2023-01-01	2024-12-01
14	Media	40000.00	2021-02-01	[null]
15	Agriculture	30000.00	2020-03-01	2023-03-01
16	Information Security	90000.00	2023-04-01	2025-04-01
17	Pharmaceuticals	80000.00	2021-05-01	[null]
18	Aerospace	75000.00	2022-06-01	2024-06-01
19	Biotechnology	85000.00	2023-07-01	2025-07-01
- Message Bar:** Shows "Total rows: 100 of 100 Query complete 00:00:00.088".
- System Bar:** Shows system status including "ENG IN", "1:35 PM", and "11/5/2024".

Education table

The screenshot shows the pgAdmin 4 interface with the following details:

- Object Explorer:** Shows the database structure under "Youngsters_db" including Schemas, Tables, and Views.
- Dashboard:** Shows the query `select * from Education;`
- Data Output:** Displays the results of the query as a table with columns: education_id, institute_id, city_id, start_date, and end_date.
- SQL:** Shows the raw SQL query used to retrieve the data.

education_id	institute_id	city_id	start_date	end_date
1	1	1	15	2019-08-01
2	2	2	34	2020-01-15
3	3	3	24	2015-06-20
4	4	4	127	2021-08-01
5	5	5	103	2020-09-01
6	6	6	8	2021-05-15
7	7	7	104	2018-07-20
8	8	8	25	2019-03-10
9	9	9	128	2021-08-01
10	10	10	105	2017-11-15
11	11	11	106	2020-02-20
12	12	12	107	2019-04-01
13	13	13	24	2016-09-01
14	14	14	106	2018-01-01
15	15	15	109	2021-02-01
16	16	16	71	2020-08-15
17	17	17	34	2015-05-01
18	18	18	34	2020-08-01
19	19	19	34	2019-09-10

Total rows: 100 of 100 Query complete 00:00:00.066

Youngster table

The screenshot shows the pgAdmin 4 interface with the following details:

- Object Explorer:** Shows the database structure under "Youngsters_db" including Schemas, Tables, and Views.
- Dashboard:** Shows the query `select * from Youngster;`
- Data Output:** Displays the results of the query as a table with columns: youngster_id, name, email, gender, education_id, employment_id, place_of_origin, and date_of_birth.
- SQL:** Shows the raw SQL query used to retrieve the data.

youngster_id	name	email	gender	education_id	employment_id	place_of_origin	date_of_birth
1	Aarav Sharma	aarav.sharma@gmail.com	Male	12	1	Delhi	1998-01-15
2	Ananya Gupta	ananya.gupta@gmail.com	Female	23	2	Mumbai	1999-02-25
3	Vihann Kumar	vihann.kumar@gmail.com	Male	34	3	Bangalore	2000-03-05
4	Aanya Verma	aanya.verma@gmail.com	Female	45	4	Chennai	1997-04-10
5	Arjun Reddy	arjun.reddy@gmail.com	Male	56	5	Hyderabad	2001-05-22
6	Pooja Singh	pooja.singh@gmail.com	Female	67	6	Kolkata	1995-06-30
7	Kabir Khan	kabir.khan@gmail.com	Male	78	7	Ahmedabad	1998-07-15
8	Diya Mehta	diya.mehta@gmail.com	Female	89	8	Surat	1999-08-12
9	Rohan Patel	rohan.patel@gmail.com	Male	92	9	Jaipur	2000-09-09
10	Meera Nair	meera.nair@gmail.com	Female	11	10	Pune	1996-10-01
11	Sai Kumar	sai.kumar@gmail.com	Male	22	11	Coimbatore	1994-11-11
12	Nisha Yadav	nisha.yadav@gmail.com	Female	33	12	Lucknow	1998-12-21
13	Ishaan Joshi	ishaan.joshi@gmail.com	Male	44	13	Kochi	2001-01-03
14	Shreya Desai	shreya.desai@gmail.com	Female	55	14	Visakhapatnam	1997-02-17
15	Akash Bansal	akash.bansal@gmail.com	Male	66	15	Nagpur	2000-03-27
16	Tara Rani	tara.rani@gmail.com	Female	77	16	Vadodara	1999-04-30
17	Karan Singh	karan.singh@gmail.com	Male	88	17	Indore	1995-05-22
18	Additi Gupta	additi.gupta@gmail.com	Female	99	18	Rajkot	1998-06-15
19	Omkar Patil	omkar.patil@gmail.com	Male	10	19	Nashik	2000-07-01

Total rows: 100 of 100 Query complete 00:00:00.070

Enrolled table

The screenshot shows the pgAdmin 4 interface with the 'Object Explorer' on the left and the 'Query Editor' on the right. The 'Object Explorer' tree shows the database structure, including 'Tables' which is currently selected. The 'Query Editor' displays the results of the query `select* from Enrolled;`. The data is presented in a table with columns: `youngster_id`, `institute_id`, `enrollment_date`, `completion_status`, `credits`, and `grade`. The table contains 19 rows of data.

youngster_id	institute_id	enrollment_date	completion_status	credits	grade
1	1	36	2020-08-15	true	30
2	2	8	2019-09-10	false	28
3	3	49	2021-01-12	true	34
4	4	10	2022-07-20	true	40
5	5	19	2018-05-25	false	20
6	6	64	2020-11-30	true	32
7	7	52	2021-02-14	true	36
8	8	27	2022-06-18	false	22
9	9	49	2019-10-11	true	30
10	10	15	2023-01-01	true	28
11	11	60	2020-03-30	false	25
12	12	30	2021-04-12	true	35
13	13	78	2019-08-21	true	31
14	14	42	2022-02-16	false	27
15	15	54	2020-12-01	true	29
16	16	70	2021-03-05	true	38
17	17	33	2018-06-29	false	21
18	18	5	2019-11-14	true	36
19	19	66	2021-08-20	true	34

Opportunities table

The screenshot shows the pgAdmin 4 interface with the 'Object Explorer' on the left and the 'Query Editor' on the right. The 'Object Explorer' tree shows the database structure, including 'Tables' which is currently selected. The 'Query Editor' displays the results of the query `select* from opportunities;`. The data is presented in a table with columns: `opportunity_id`, `opportunity_type`, `location`, `employer_id`, `salary_benefits`, and `eligibility_criteria`. The table contains 19 rows of data.

opportunity_id	opportunity_type	location	employer_id	salary_benefits	eligibility_criteria
1	1	Internship	Mumbai	₹15,000 per month	Undergraduate
2	2	Full-time	Delhi	₹60,000 per month	Postgraduate
3	3	Part-time	Bangalore	₹25,000 per month	Undergraduate
4	4	Contract	Hyderabad	₹50,000 per month	Any Graduate
5	5	Freelance	Chennai	Project-based	Experience preferred
6	6	Internship	Kolkata	₹12,000 per month	Undergraduate
7	7	Full-time	Pune	₹70,000 per month	Postgraduate
8	8	Part-time	Ahmedabad	₹20,000 per month	Diploma
9	9	Contract	Jaipur	₹55,000 per month	Postgraduate
10	10	Freelance	Lucknow	Project-based	Experience preferred
11	11	Internship	Surat	₹18,000 per month	Undergraduate
12	12	Full-time	Nagpur	₹62,000 per month	Postgraduate
13	13	Part-time	Coimbatore	₹22,000 per month	Any Graduate
14	14	Contract	Vishakhapatnam	₹48,000 per month	Experience preferred
15	15	Freelance	Vijayawada	Project-based	Undergraduate
16	16	Internship	Indore	₹14,000 per month	Undergraduate
17	17	Full-time	Mysore	₹68,000 per month	Postgraduate
18	18	Part-time	Nashik	₹21,000 per month	Any Graduate
19	19	Contract	Aurangabad	₹53,000 per month	Postgraduate

Migration_Event table

The screenshot shows the pgAdmin 4 interface with the 'Tables' node selected in the Object Explorer. The main query editor displays the following SQL query and its results:

```
select * from opportunities;
```

opportunity_id	opportunity_type	location	employer_id	salary_benefits	eligibility_criteria
1	1	Mumbai	56	₹15,000 per month	Undergraduate
2	2	Delhi	23	₹60,000 per month	Postgraduate
3	3	Bangalore	89	₹25,000 per month	Undergraduate
4	4	Hyderabad	45	₹50,000 per month	Any Graduate
5	5	Chennai	78	Project-based	Experience preferred
6	6	Kolkata	34	₹12,000 per month	Undergraduate
7	7	Pune	91	₹70,000 per month	Postgraduate
8	8	Ahmedabad	10	₹20,000 per month	Diploma
9	9	Jaipur	82	₹55,000 per month	Postgraduate
10	10	Lucknow	37	Project-based	Experience preferred
11	11	Surat	66	₹18,000 per month	Undergraduate
12	12	Nagpur	53	₹62,000 per month	Postgraduate
13	13	Coimbatore	99	₹22,000 per month	Any Graduate
14	14	Vishakhapatnam	18	₹48,000 per month	Experience preferred
15	15	Vijayawada	84	Project-based	Undergraduate
16	16	Indore	30	₹14,000 per month	Undergraduate
17	17	Mysore	72	₹68,000 per month	Postgraduate
18	18	Nashik	4	₹21,000 per month	Any Graduate
19	19	Aurangabad	60	₹33,000 per month	Postgraduate

Total rows: 162 of 162 Query complete 00:00:00.090 Ln 1, Col 27

Youngster_phone table

The screenshot shows the pgAdmin 4 interface with the 'Tables' node selected in the Object Explorer. The main query editor displays the following SQL query and its results:

```
select * from youngster_phone;
```

youngster_id	phone_no
1	9876543210
2	9123456789
3	9876543211
4	9876543212
5	9123456790
6	9876543213
7	9876543214
8	9876543215
9	9876543216
10	9876543217
11	9876543218
12	9876543219
13	9876543220
14	9876543221
15	9876543222
16	9876543223
17	9876543224
18	9876543225
19	9876543226

Total rows: 102 of 102 Query complete 00:00:00.059 Ln 1, Col 29

4.3 SQL Queries

QUERIES

1) Retrieve all cities along with their corresponding state names.

```
SELECT City.City_Name, State.State_Name  
FROM City  
JOIN State ON City.State_ID = State.State_ID;
```

The screenshot shows the pgAdmin 4 interface with a query editor and a results grid. The query editor contains the following SQL code:

```
--1 Retrieve all cities along with their corresponding state names.  
SELECT City.City_Name, State.State_Name  
FROM City  
JOIN State ON City.State_ID = State.State_ID;
```

The results grid displays 141 rows of data, mapping cities to their respective states:

	city_name	state_name
1	Vizagapatnam	Andhra Pradesh
2	Vijayawada	Andhra Pradesh
3	Itanagar	Arunachal Pradesh
4	Guwahati	Assam
5	Patna	Bihar
6	Rajpur	Chhattisgarh
7	Panaji	Goa
8	Ahmedabad	Gujarat
9	Gurugram	Haryana
10	Shimla	Himachal Pradesh
11	Ranchi	Jharkhand
12	Bengaluru	Karnataka
13	Thiruvananthapuram	Kerala
14	Bhopal	Madhya Pradesh

Total rows: 141 of 141 Query complete 00:00:00.054 Ln 1754, Col

Number of tuples: 141

2) Health facilities based on their type

```
SELECT * FROM Health_Facility  
WHERE Facility_Type = 'Hospital';
```

The screenshot shows the pgAdmin 4 interface with a query editor window. The query is:

```
--1 Retrieve all cities along with their corresponding state names.  
--2 Health facilities based on their type  
SELECT * FROM Health_Facility  
WHERE Facility_Type = 'Hospital';
```

The results table displays 9 rows of data:

facility_id	name	facility_type	bed_capacity
1	All India Institute of Medical Sciences	Hospital	800
2	Post Graduate Institute of Medical Education and Resear...	Hospital	1500
3	Sri Ramachandra Medical Centre	Hospital	700
4	Sitaram Bhartia Institute of Science and Research	Hospital	100
5	Sri Aurobindo Institute of Medical Sciences	Hospital	400
6	Vijaya Medical Centre	Hospital	250
7	St. John's Medical College Hospital	Hospital	500
8	St. Thomas Hospital	Hospital	400
9	Kasturba Hospital	Hospital	600

Total rows: 9 of 9 Query complete 00:00:00.115 Ln 1749, Col 1

Number of tuples: 9

3) Filter cities by a specific climate type.

```
SELECT City.City_Name, Climate.Climate_Type  
FROM City  
JOIN Climate ON City.City_ID = Climate.City_ID  
WHERE Climate.Climate_Type = 'Tropical';
```

The screenshot shows a SQL query editor interface. The top pane displays the SQL code for filtering cities by climate type:

```
--3 Filter cities by a specific climate type.  
SELECT DISTINCT City.City_Name, Climate.Climate_Type  
FROM City  
JOIN Climate ON City.City_ID = Climate.City_ID  
WHERE Climate.Climate_Type = 'Tropical'  
ORDER BY City_Name;
```

The bottom pane shows the resulting data table:

	city_name	climate_type
1	Bhopal	Tropical
2	Chennai	Tropical
3	Delhi	Tropical
4	Diu	Tropical
5	Gurugram	Tropical
6	Imphal	Tropical
7	Jaipur	Tropical
8	Kohima	Tropical
9	Kolkata	Tropical
10	Lucknow	Tropical
11	Raipur	Tropical
12	Ranchi	Tropical
13	Vijayawada	Tropical
14	Visakhapatnam	Tropical

Total rows: 14 of 14 Query complete 00:00:00.080 Ln 1764, Col 19

Number of tuples: 14

4) Find opportunities from employers.

```
SELECT Employer.Employer_Name,  
Opportunities.Opportunity_Type  
FROM Employer  
JOIN Opportunities ON Employer.Employer_ID =  
Opportunities.Employer_ID;
```

The screenshot shows a SQL query editor interface. The top pane displays the SQL code for selecting employer names and opportunity types, joining the Employer and Opportunities tables based on Employer_ID. The bottom pane shows the resulting data output as a table with two columns: employer_name and opportunity_type. The data includes 75 rows of employer names like NTPC, Zomato, PayPal, etc., and their corresponding opportunity types such as Internship, Full-time, Part-time, Contract, Freelance, etc.

employer_name	opportunity_type
1 NTPC	Internship
2 Zomato	Full-time
3 PayPal	Part-time
4 Tata Motors	Contract
5 Air India	Freelance
6 Hindustan Unilever	Internship
7 Razorpay	Full-time
8 Oracle	Part-time
9 Reliance Jio	Contract
10 Godrej	Freelance
11 ITC Hotels	Internship
12 ISRO	Full-time
13 Practo	Part-time
14 Adobe	Contract

Number of tuples: 75

5) Count cities grouped by state.

```
SELECT State.State_Name, COUNT(City.City_ID) AS  
City_Count  
FROM State  
JOIN City ON State.State_ID = City.State_ID  
GROUP BY State.State_Name;
```

The screenshot shows a SQL query being run in a database environment. The query is:

```
--5 Count cities grouped by state.  
1772 ✓ SELECT State.State_Name, COUNT(City.City_ID) AS City_Count  
1773 FROM State  
1774 JOIN City ON State.State_ID = City.State_ID  
1775 GROUP BY State.State_Name;  
1776
```

The results are displayed in a table:

	state_name	city_count
1	Uttarakhand	3
2	Himachal Pradesh	6
3	Assam	2
4	Andhra Pradesh	8
5	Madhya Pradesh	6
6	Sikkim	2
7	Jammu and Kashmir	4
8	Goa	1
9	Meghalaya	1
10	Dadra and Nagar Haveli and Daman and Diu	1
11	Maharashtra	17
12	Puducherry	1
13	Mizoram	1
14	Andaman and Nicobar Islands	1
15	Haryana	3

Total rows: 35 of 35 Query complete 00:00:00.065 Ln 1776, Col 1

Number of tuples: 35

6) Group and count youngsters by gender.

```
SELECT Gender, COUNT(Youngster_ID)
FROM Youngster GROUP BY Gender;
```

The screenshot shows a SQL query window with the following details:

- Query History:** Shows the executed query:

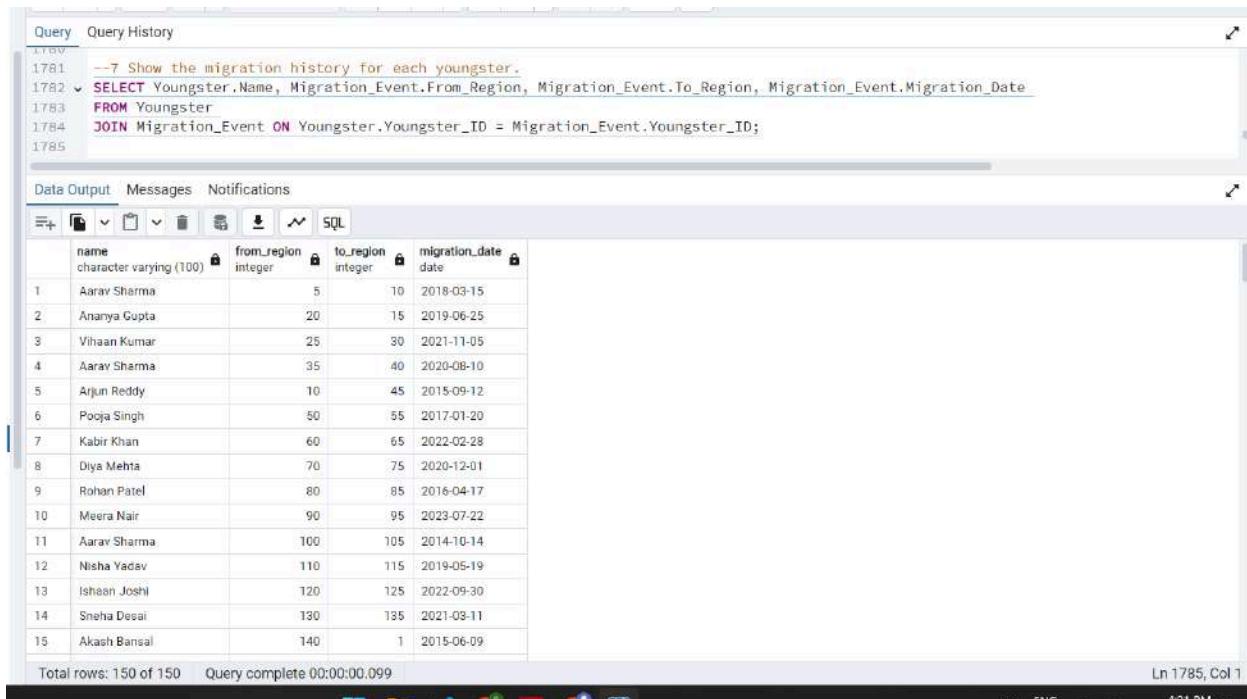
```
-- 6. Group and count youngsters by gender.
SELECT Gender, COUNT(Youngster_ID)
FROM Youngster GROUP BY Gender;
```
- Data Output:** Displays the results of the query in a table:

	gender	count
1	Female	50
2	Male	50
- Status Bar:** Shows "Total rows: 2 of 2" and "Query complete 00:00:00.064".
- System Tray:** Shows system icons including a battery icon at 4:16 PM on 11/2/2024.

Number of tuples: 2

7) Show the migration history for each youngster.

```
SELECT Youngster.Name, Migration_Event.From_Region,  
Migration_Event.To_Region, Migration_Event.Migration_Date  
FROM Youngster  
JOIN Migration_Event ON Youngster.Youngster_ID =  
Migration_Event.Youngster_ID;
```



The screenshot shows a SQL query window with the following details:

- Query History:** Shows the executed SQL code:

```
--7 Show the migration history for each youngster.  
SELECT Youngster.Name, Migration_Event.From_Region, Migration_Event.To_Region, Migration_Event.Migration_Date  
FROM Youngster  
JOIN Migration_Event ON Youngster.Youngster_ID = Migration_Event.Youngster_ID;
```
- Data Output:** A table displaying the results of the query. The table has columns: name, from_region, to_region, and migration_date. The data consists of 150 rows, each representing a migration event for a youngster. The first few rows are:

	name	from_region	to_region	migration_date
1	Aarav Sharma	5	10	2018-03-15
2	Ananya Gupta	20	15	2019-06-25
3	Vihann Kumar	25	30	2021-11-05
4	Aarav Sharma	35	40	2020-08-10
5	Arjun Reddy	10	45	2015-09-12
6	Pooja Singh	50	55	2017-01-20
7	Kabir Khan	60	65	2022-02-28
8	Diya Mehta	70	75	2020-12-01
9	Rohan Patel	80	85	2016-04-17
10	Meera Nair	90	95	2023-07-22
11	Aarav Sharma	100	105	2014-10-14
12	Nisha Yadav	110	115	2019-05-19
13	Ishaan Joshi	120	125	2022-09-30
14	Sneha Desai	130	135	2021-03-11
15	Akash Bansal	140	1	2015-06-09
- Bottom Status Bar:** Shows "Total rows: 150 of 150" and "Query complete 00:00:00.099".
- System Status:** Shows "Ln 1785, Col 1" and the system tray with icons for battery, signal, and time (4:21 PM).

Number of tuples: 150

8)Display opportunities in a specific location.

```
SELECT Opportunity_Type, Salary_Benefits  
FROM Opportunities  
WHERE Location = 'Ahmedabad';
```

The screenshot shows a PostgreSQL client interface with the following details:

- Query History:** Displays the executed SQL query:

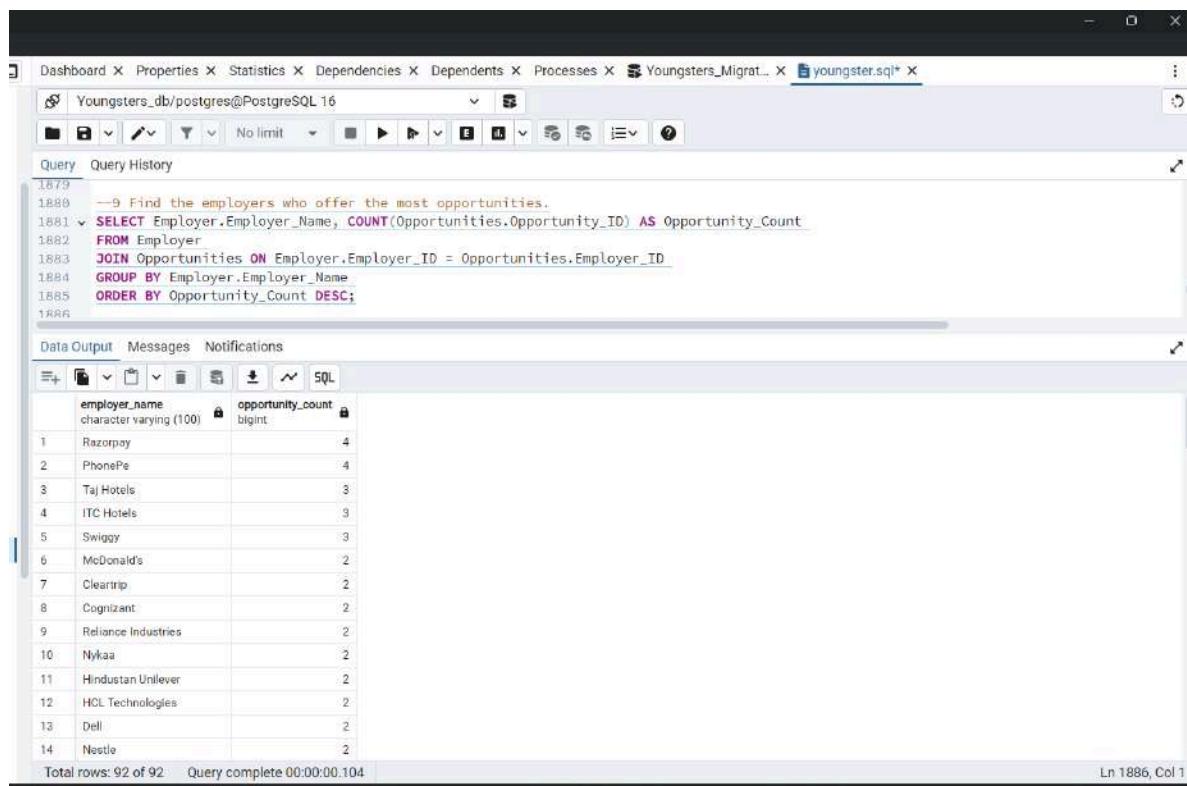
```
1874  
1875 --8 Display opportunities in a specific location.  
1876 v SELECT Opportunity_Type, Salary_Benefits  
1877   FROM Opportunities  
1878   WHERE Location = 'Ahmedabad';  
1879
```
- Data Output:** Shows the result of the query as a table:

	opportunity_type	salary_benefits
1	Part-time	₹20,000 per month
2	Part-time	₹19,500 per month
3	Part-time	₹19,200 per month

Number of tuples: 3

9) Find the employers who offer the most opportunities.

```
SELECT Employer.Employer_Name,  
COUNT(Opportunities.Opportunity_ID) AS Opportunity_Count  
FROM Employer  
JOIN Opportunities ON Employer.Employer_ID =  
Opportunities.Employer_ID  
GROUP BY Employer.Employer_Name  
ORDER BY Opportunity_Count DESC;
```



The screenshot shows a PostgreSQL client interface with the following details:

- Query History:** Displays the executed SQL code:

```
1879  
1880 --9 Find the employers who offer the most opportunities.  
1881 v SELECT Employer.Employer_Name, COUNT(Opportunities.Opportunity_ID) AS Opportunity_Count  
1882 FROM Employer  
1883 JOIN Opportunities ON Employer.Employer_ID = Opportunities.Employer_ID  
1884 GROUP BY Employer.Employer_Name  
1885 ORDER BY Opportunity_Count DESC;  
1886
```
- Data Output:** Shows a table with two columns: **employer_name** and **opportunity_count**. The data is as follows:

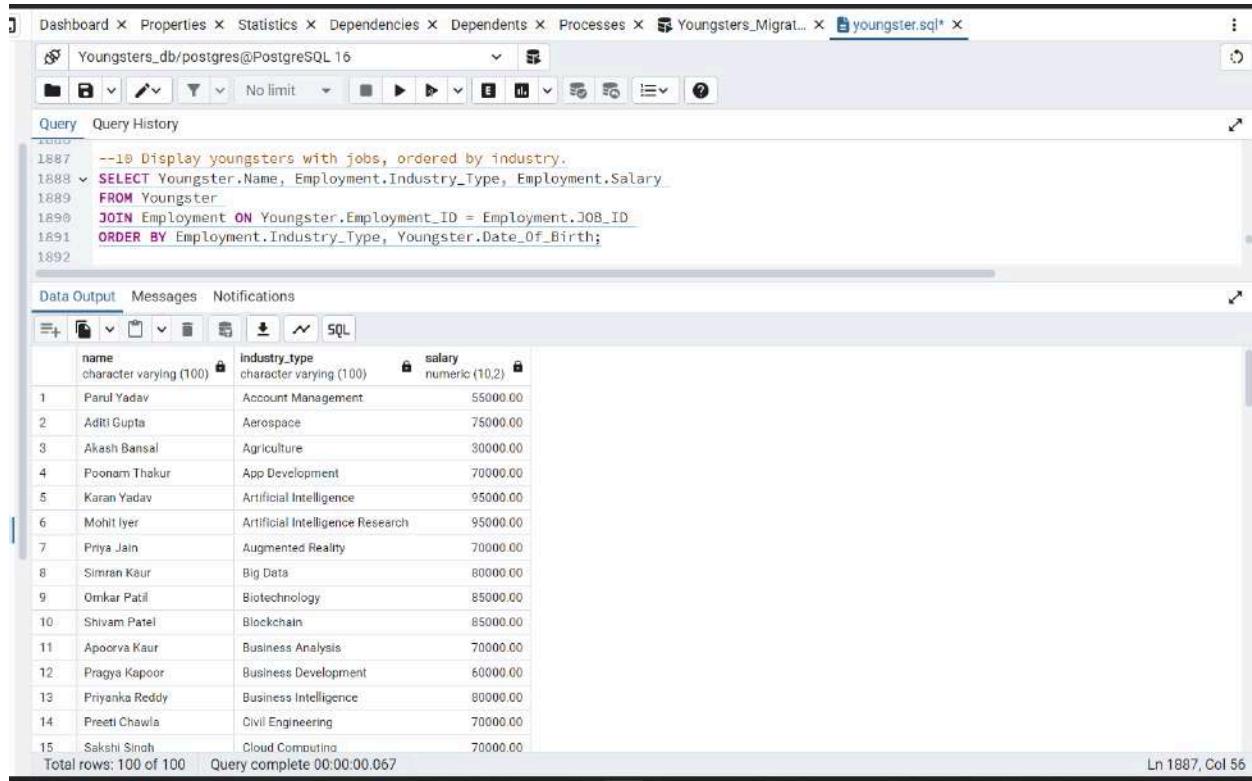
	employer_name	opportunity_count
1	Razorpay	4
2	PhonePe	4
3	Taj Hotels	3
4	ITC Hotels	3
5	Swiggy	3
6	McDonald's	2
7	Cleartrip	2
8	Cognizant	2
9	Reliance Industries	2
10	Nykaa	2
11	Hindustan Unilever	2
12	HCL Technologies	2
13	Dell	2
14	Nestle	2

Total rows: 92 Query complete 00:00:00.104 Ln 1886, Col 1

Number of tuples: 92

10) Display youngsters with jobs, ordered by industry.

```
SELECT Youngster.Name, Employment.Industry_Type,  
Employment.Salary  
FROM Youngster  
JOIN Employment ON Youngster.Employment_ID =  
Employment.JOB_ID  
ORDER BY Employment.Industry_Type,  
Youngster.DateOfBirth;
```



The screenshot shows the pgAdmin 4 interface with the following details:

- Toolbar:** Includes tabs for Dashboard, Properties, Statistics, Dependencies, Dependents, Processes, and a file tab for "Youngsters_Migrat...".
- Query Editor:** Shows the SQL code for the query.
- Data Output:** Displays the results of the query, which consists of three columns: name, industry_type, and salary. The results are as follows:

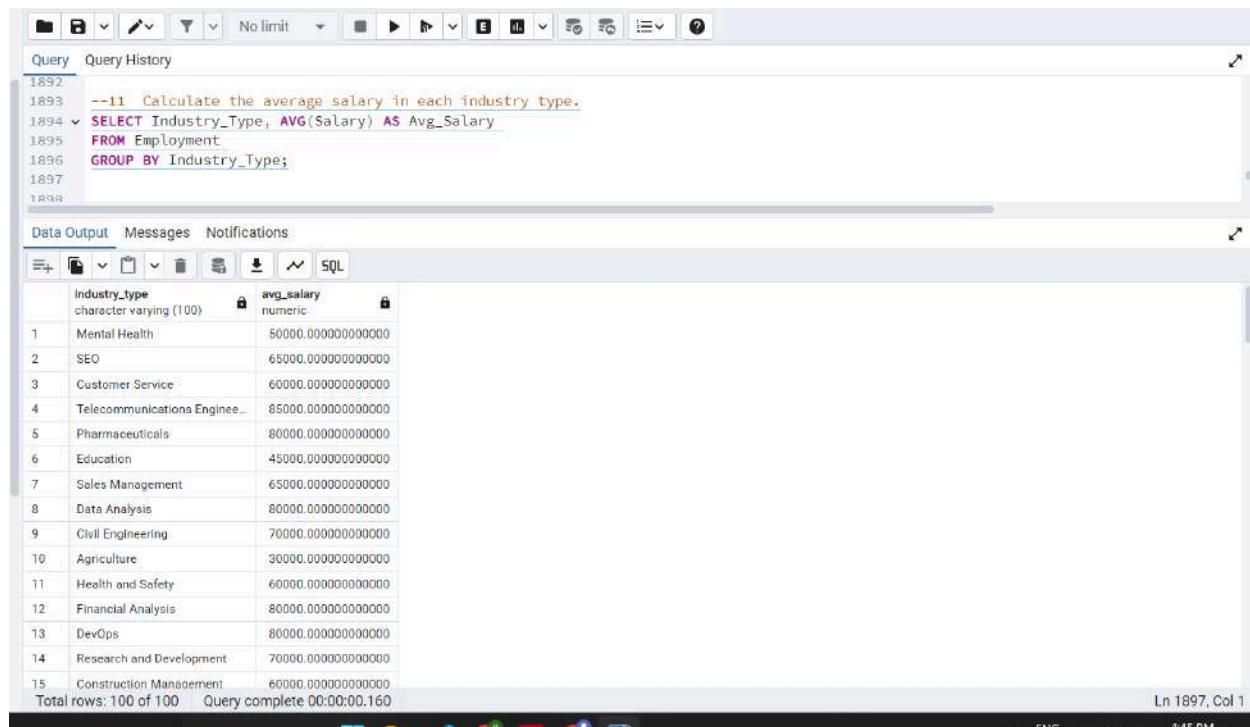
	name	industry_type	salary
1	Parul Yadav	Account Management	55000.00
2	Aditi Gupta	Aerospace	75000.00
3	Akash Bansal	Agriculture	30000.00
4	Poonam Thakur	App Development	70000.00
5	Karan Yadav	Artificial Intelligence	95000.00
6	Mohit Iyer	Artificial Intelligence Research	95000.00
7	Priya Jain	Augmented Reality	70000.00
8	Simran Kaur	Big Data	80000.00
9	Omkar Patil	Biotechnology	85000.00
10	Shivam Patel	Blockchain	85000.00
11	Apoorva Kaur	Business Analysis	70000.00
12	Pragya Kapoor	Business Development	60000.00
13	Priyanka Reddy	Business Intelligence	80000.00
14	Preeti Chawla	Civil Engineering	70000.00
15	Sakshi Singh	Cloud Computing	70000.00

Total rows: 100 of 100 Query complete 00:00:00.067 Ln 1887, Col 56

Number of tuples: 100

11) Calculate the average salary in each industry type.

```
SELECT Industry_Type, AVG(Salary) AS Avg_Salary  
FROM Employment  
GROUP BY Industry_Type;
```



The screenshot shows a SQL query editor interface. The top pane displays the SQL code:

```
--11 Calculate the average salary in each industry type.  
SELECT Industry_Type, AVG(Salary) AS Avg_Salary  
FROM Employment  
GROUP BY Industry_Type;
```

The bottom pane shows the results of the query, which is a table with two columns: 'Industry_type' and 'avg_salary'. The table contains 15 rows of data, corresponding to the 15 industry types listed in the question. The data is as follows:

	Industry_type	avg_salary
1	Mental Health	50000.000000000000
2	SEO	65000.000000000000
3	Customer Service	60000.000000000000
4	Telecommunications Enginee...	85000.000000000000
5	Pharmaceuticals	80000.000000000000
6	Education	45000.000000000000
7	Sales Management	65000.000000000000
8	Data Analysis	80000.000000000000
9	Civil Engineering	70000.000000000000
10	Agriculture	30000.000000000000
11	Health and Safety	60000.000000000000
12	Financial Analysis	80000.000000000000
13	DevOps	80000.000000000000
14	Research and Development	70000.000000000000
15	Construction Management	60000.000000000000

Total rows: 100 of 100 Query complete 00:00:00.160

Ln 1897, Col 1

Number of tuples: 100

12) List all youngsters and their phone numbers.

```
SELECT Youngster.Name, Youngster_Phone.Phone_no  
FROM Youngster  
JOIN Youngster_Phone ON Youngster.Youngster_ID =  
Youngster_Phone.Youngster_ID;
```

The screenshot shows the pgAdmin interface with a query editor and a data output viewer.

Query Editor:

```
--12 List all youngsters and their phone numbers.  
SELECT Youngster.Name, Youngster_Phone.Phone_no  
FROM Youngster  
JOIN Youngster_Phone ON Youngster.Youngster_ID = Youngster_Phone.Youngster_ID;
```

Data Output:

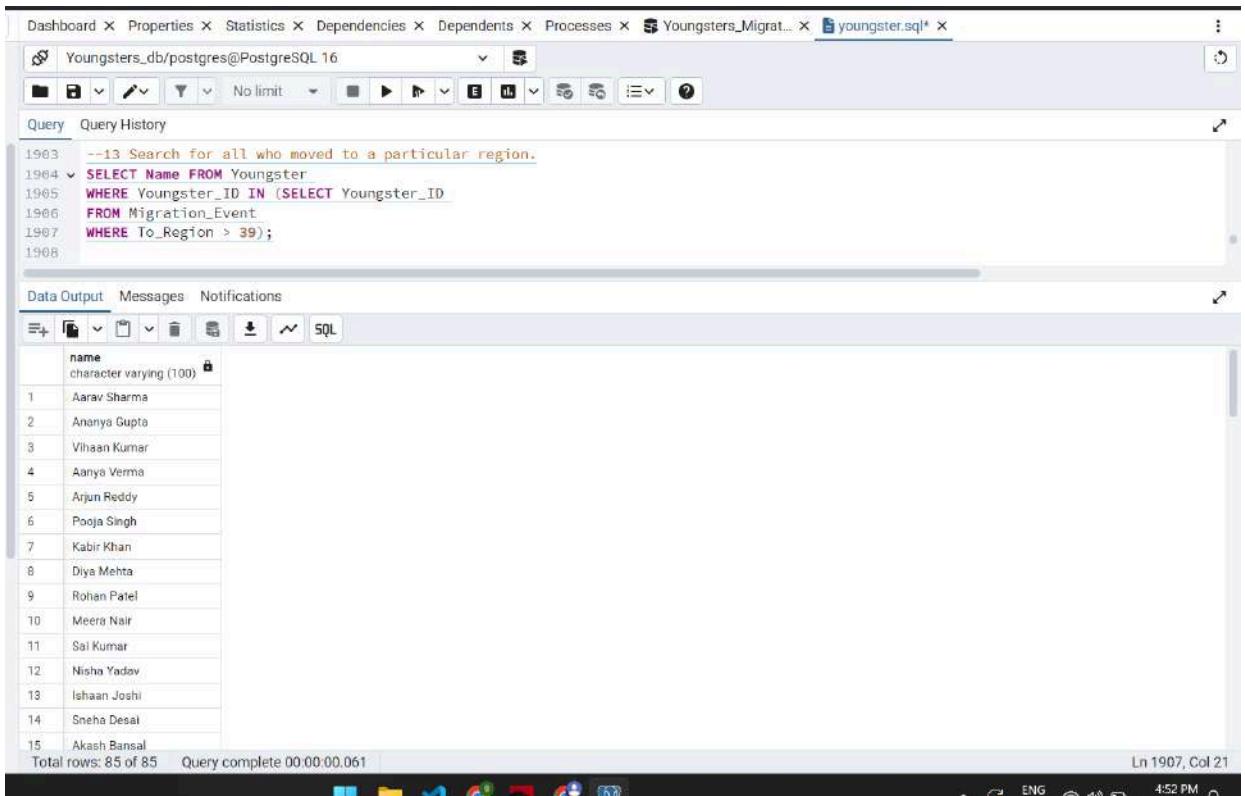
	name	phone_no
1	Aarav Sharma	9876543210
2	Aarav Sharma	9123456789
3	Ananya Gupta	9876543211
4	Vihaan Kumar	9876543212
5	Vihaan Kumar	9123456790
6	Aanya Verma	9876543213
7	Arjun Reddy	9876543214
8	Pooja Singh	9876543215
9	Kabir Khan	9876543216
10	Diya Mehta	9876543217
11	Rohan Patel	9876543218
12	Meera Nair	9876543219
13	Sal Kumar	9876543220
14	Nisha Yadav	9876543221
15	Ishaan Joshi	9876543222

Total rows: 102 Query complete 00:00:00.055 Ln 1902, Col 1

Number of tuples: 102

13) Search for all who moved to particular regions.

```
SELECT Name FROM Youngster  
WHERE Youngster_ID IN (SELECT Youngster_ID  
FROM Migration_Event  
WHERE To_Region > 39);
```



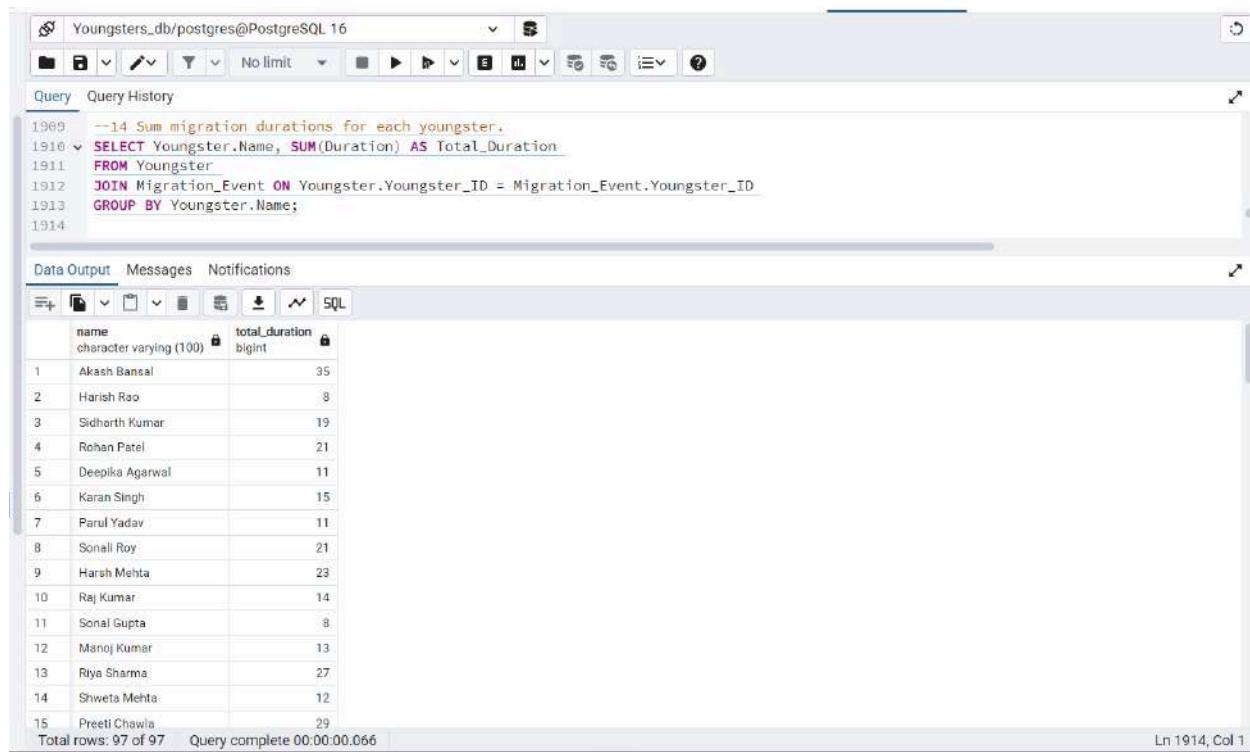
The screenshot shows the pgAdmin 4 interface with the following details:

- Toolbar:** Dashboard, Properties, Statistics, Dependencies, Dependents, Processes, Youngsters_Migrat..., youngster.sql*
- Query Editor:** Shows the SQL query from the previous code block.
- Data Output:** A table titled "name" with 85 rows, listing names such as Aarav Sharma, Ananya Gupta, Vihaan Kumar, Aanya Verma, Arjun Reddy, Pooja Singh, Kabir Khan, Diya Mehta, Rohan Patel, Meera Nair, Sai Kumar, Nisha Yadav, Ishaan Joshi, Sneha Desai, and Akash Bansal.
- Bottom Status:** Total rows: 85 of 85, Query complete 00:00:00.061, Ln 1907, Col 21

Number of tuples: 85

14) Sum migration durations for each youngster.

```
SELECT Youngster.Name, SUM(Duration) AS Total_Duration  
FROM Youngster  
JOIN Migration_Event ON Youngster.Youngster_ID =  
Migration_Event.Youngster_ID  
GROUP BY Youngster.Name;
```



The screenshot shows a PostgreSQL client interface with the following details:

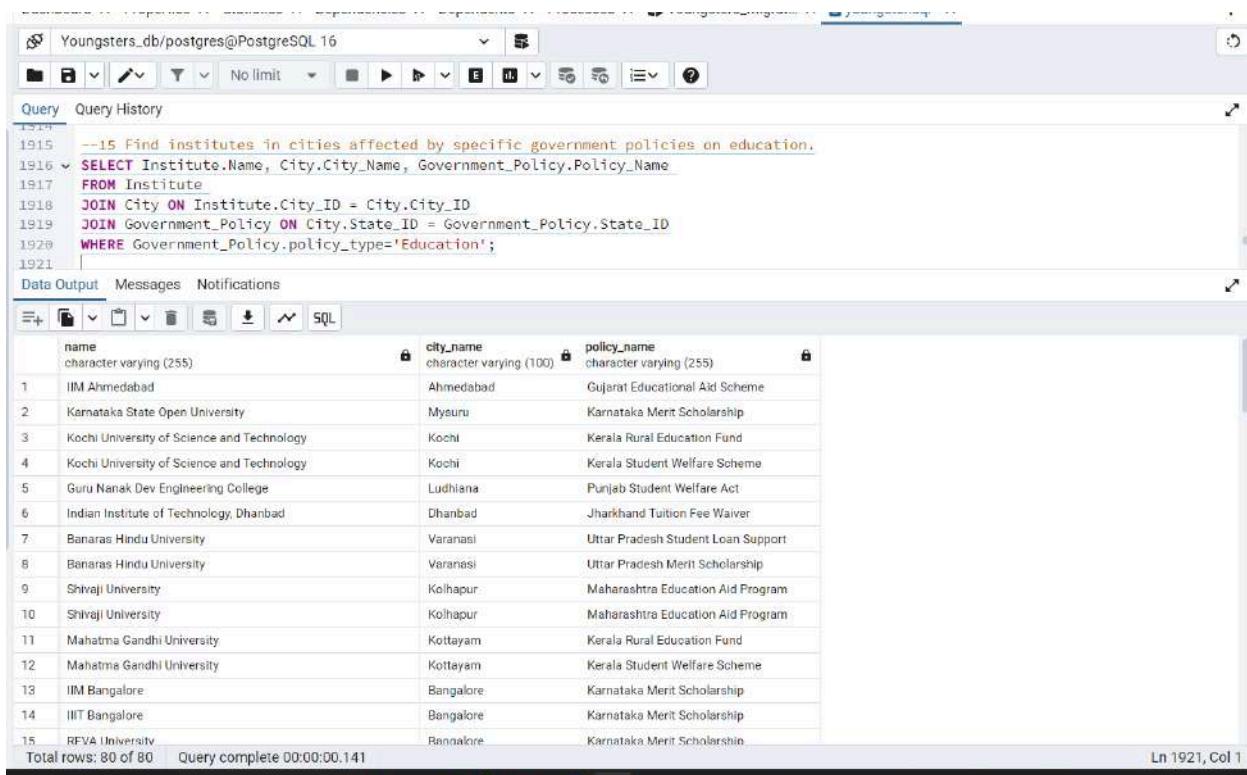
- Query History:** Contains the SQL code for question 14.
- Data Output:** Displays the results of the query as a table.
- Table Data:**

	name	total_duration
1	Akash Bansal	35
2	Harish Rao	8
3	SidhARTH Kumar	19
4	Rohan Patel	21
5	Deepika Agarwal	11
6	Karan Singh	15
7	Parul Yadav	11
8	Sonali Roy	21
9	Harsh Mehta	23
10	Raj Kumar	14
11	Sonal Gupta	8
12	Manoj Kumar	13
13	Riya Sharma	27
14	Shweta Mehta	12
15	Preeti Chawla	29
- Total rows:** 97
- Query complete:** 00:00:00.066
- Line number:** Ln 1914, Col 1

Number of tuples: 97

15) Find institutes in cities affected by specific government policies on education.

```
SELECT Institute.Name, City.City_Name,  
Government_Policy.Policy_Name  
FROM Institute  
JOIN City ON Institute.City_ID = City.City_ID  
JOIN Government_Policy ON City.State_ID =  
Government_Policy.State_ID  
WHERE Government_Policy.policy_type='Education';
```



The screenshot shows the pgAdmin interface with a query editor and a data output viewer.

Query Editor:

```
15. --15 Find institutes in cities affected by specific government policies on education.  
16. v SELECT Institute.Name, City.City_Name, Government_Policy.Policy_Name  
17. FROM Institute  
18. JOIN City ON Institute.City_ID = City.City_ID  
19. JOIN Government_Policy ON City.State_ID = Government_Policy.State_ID  
20. WHERE Government_Policy.policy_type='Education';  
21.
```

Data Output:

	name	city_name	policy_name
1	IIM Ahmedabad	Ahmedabad	Gujarat Educational Aid Scheme
2	Karnataka State Open University	Mysuru	Karnataka Merit Scholarship
3	Kochi University of Science and Technology	Kochi	Kerala Rural Education Fund
4	Kochi University of Science and Technology	Kochi	Kerala Student Welfare Scheme
5	Guru Nanak Dev Engineering College	Ludhiana	Punjab Student Welfare Act
6	Indian Institute of Technology, Dhanbad	Dhanbad	Jharkhand Tuition Fee Waiver
7	Banaras Hindu University	Varanasi	Uttar Pradesh Student Loan Support
8	Banaras Hindu University	Varanasi	Uttar Pradesh Merit Scholarship
9	Shivaji University	Kolhapur	Maharashtra Education Aid Program
10	Shivaji University	Kolhapur	Maharashtra Education Aid Program
11	Mahatma Gandhi University	Kottayam	Kerala Rural Education Fund
12	Mahatma Gandhi University	Kottayam	Kerala Student Welfare Scheme
13	IIM Bangalore	Bangalore	Karnataka Merit Scholarship
14	IIT Bangalore	Bangalore	Karnataka Merit Scholarship
15	RFVA University	Bangalore	Karnataka Merit Scholarship

Total rows: 80 of 80 Query complete 00:00:00.141 Ln 1921, Col 1

Number of tuples: 80

16) Calculate the average temperature per region

```
SELECT c.Climate_Type, ci.City_Name, AVG(c.Average_Temp)
AS Avg_Temperature
FROM Climate c
JOIN City ci ON c.City_ID = ci.City_ID
GROUP BY c.Climate_Type, ci.City_Name
ORDER BY c.Climate_Type, ci.City_Name;
```

The screenshot shows a PostgreSQL client interface with the following details:

- Query Tab:** Contains the SQL code for calculating average temperature by region.
- Data Output Tab:** Displays the results of the query in a table format.
- Table Headers:** climate_type, city_name, avg_temperature.
- Table Data:** 14 rows of data, showing various cities and their average temperatures across different climate types.
- Total Rows:** 33 of 33
- Query Complete:** 00:00:00.066
- Line Number:** Ln 1927, Col 38

climate_type	city_name	avg_temperature
Cold	Ahmedabad	11.166666666666667
Cold	Bengaluru	9.333333333333333
Cold	Chandigarh	11.000000000000000
Cold	Hyderabad	8.000000000000000
Cold	Mumbai	11.166666666666667
Cold	Patna	8.166666666666667
Cold	Port Blair	9.166666666666667
Cold	Shillong	8.333333333333333
Mediterranean	Agartala	15.000000000000000
Mediterranean	Bhubaneswar	14.333333333333333
Mediterranean	Kavaratti	12.000000000000000
Mediterranean	Panaji	15.333333333333333
Temperate	Aizawl	19.166666666666667
Temperate	Dehradun	13.166666666666667

Number of tuples: 33

17) Eligible youngsters for specific opportunities

```
SELECT DISTINCT y.Name, o.opportunity_type, o.location
FROM Youngster y
JOIN Opportunities o ON o.Eligibility_Criteria LIKE
'Undergraduate'
WHERE y.Education_ID IS NOT NULL AND
o.Opportunity_Type ='Internship';
```

The screenshot shows the pgAdmin interface with a query editor window. The query is:

```
--17_Eligible youngsters for specific opportunities
SELECT DISTINCT y.Name, o.opportunity_type, o.location
FROM Youngster y
JOIN Opportunities o ON o.eligibility_criteria LIKE 'Undergraduate'
WHERE y.education_id IS NOT NULL AND o.opportunity_type = 'Internship';
```

The results table has three columns: name, opportunity_type, and location. The data is as follows:

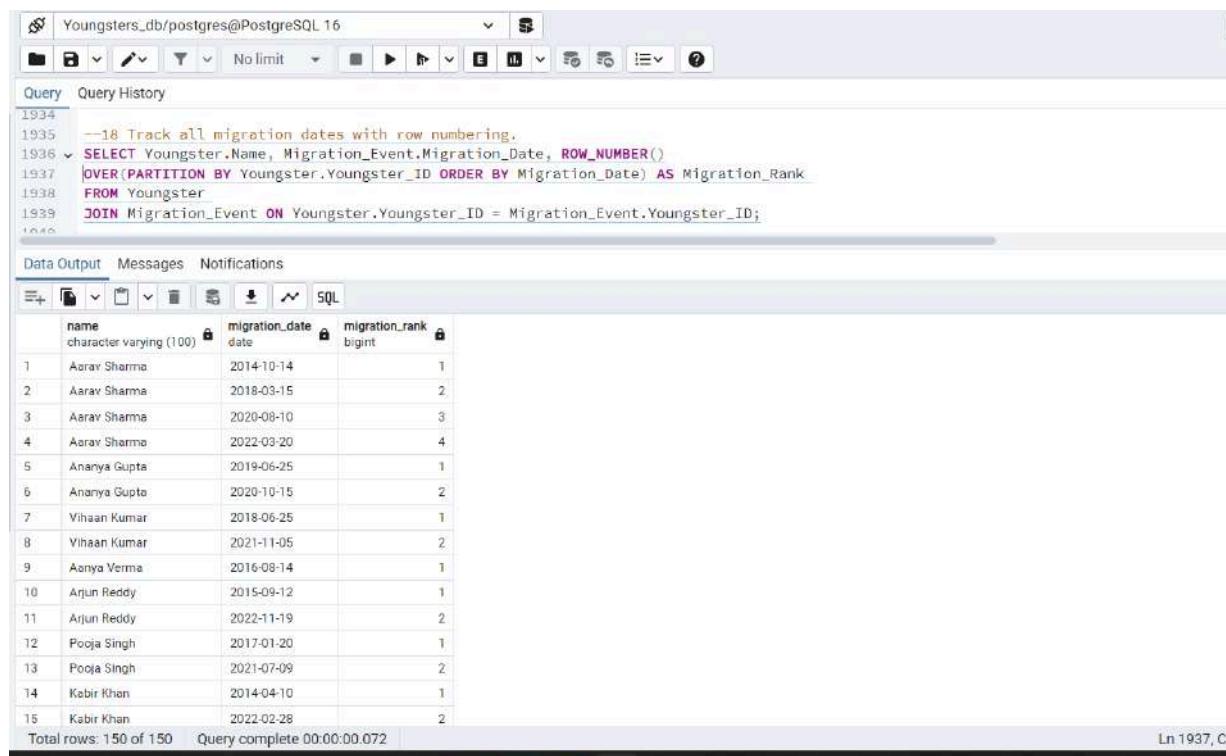
	name	opportunity_type	location
1	Ravindra Singh	Internship	Thiruvananthapuram
2	Sonal Gupta	Internship	Neyyattinkara
3	Harsh Mehta	Internship	Indore
4	Vishakha Patel	Internship	Mumbai
5	Sakshi Singh	Internship	Surat
6	Vivek Reddy	Internship	Agartala
7	Sandeep Verma	Internship	Ernakulam
8	Naina Gupta	Internship	Indore
9	Omkar Patil	Internship	Kolkata
10	Arjun Reddy	Internship	Vadodara
11	Riya Mehta	Internship	Thiruvananthapuram
12	Apoorva Kaur	Internship	Neyyattinkara
13	Preeti Chawla	Internship	Thiruvananthapuram
14	Aakash Iyer	Internship	Puducherry
15	Gaurav Yadav	Internship	Thiruvananthapuram

Total rows: 1000 of 1455 Query complete 00:00:00.101 Ln 1932, Col 54

Number of tuples 1455

18) Track all migration dates with row numbering.

```
SELECT Youngster.Name, Migration_Event.Migration_Date,  
ROW_NUMBER()  
OVER(PARTITION BY Youngster.Youngster_ID ORDER BY  
Migration_Date) AS Migration_Rank  
FROM Youngster  
JOIN Migration_Event ON Youngster.Youngster_ID =  
Migration_Event.Youngster_ID;
```



The screenshot shows a PostgreSQL client interface with the following details:

- Query History:** The history contains the following entries:
 - 1934: --18 Track all migration dates with row numbering.
 - 1935: SELECT Youngster.Name, Migration_Event.Migration_Date, ROW_NUMBER()
 - 1936: OVER(PARTITION BY Youngster.Youngster_ID ORDER BY Migration_Date) AS Migration_Rank
 - 1937: FROM Youngster
 - 1938: JOIN Migration_Event ON Youngster.Youngster_ID = Migration_Event.Youngster_ID;
 - 1939: (This entry is partially visible)
- Data Output:** The results table has three columns: name, migration_date, and migration_rank. The data is as follows:

	name	migration_date	migration_rank
1	Aarav Sharma	2014-10-14	1
2	Aarav Sharma	2018-03-15	2
3	Aarav Sharma	2020-08-10	3
4	Aarav Sharma	2022-03-20	4
5	Ananya Gupta	2019-06-25	1
6	Ananya Gupta	2020-10-15	2
7	Vihaan Kumar	2018-06-25	1
8	Vihaan Kumar	2021-11-05	2
9	Aonya Verma	2016-08-14	1
10	Arjun Reddy	2015-09-12	1
11	Arjun Reddy	2022-11-19	2
12	Pooja Singh	2017-01-20	1
13	Pooja Singh	2021-07-09	2
14	Kabir Khan	2014-04-10	1
15	Kabir Khan	2022-02-28	2

Total rows: 150 of 150 Query complete 00:00:00.072 Ln 1937, Col 1

Number of tuples: 150

19) Find youngsters with the longest single employment.

```
SELECT Name, MAX(Employment.End_Date - Employment.Start_Date) AS Employment_Time  
FROM Youngster  
JOIN Employment ON Youngster.Employment_ID = Employment.JOB_ID  
GROUP BY Name  
ORDER BY Employment_Time;
```

The screenshot shows a PostgreSQL client interface with the following details:

- Toolbar:** Includes tabs for Dashboard, Properties, Statistics, Dependencies, Dependents, Processes, and a file named "Youngsters_Migrat...".
- Query History:** Shows the executed SQL code:

```
--19 Find youngsters with their employment time.  
SELECT Name, MAX(Employment.End_Date - Employment.Start_Date) AS Employment_Time  
FROM Youngster  
JOIN Employment ON Youngster.Employment_ID = Employment.JOB_ID  
GROUP BY Name  
ORDER BY Employment_Time;
```
- Data Output:** A table showing the results of the query:

	name	employment_time
1	Raj Kumar	366
2	Revi Kumar	366
3	Neha Bansal	366
4	Ishaan Joshi	700
5	Preeti Chawla	730
6	Simran Khanna	730
7	Rohit Yadav	731
8	Karan Yadav	731
9	Omkar Patil	731
10	Kabir Khan	731
11	Rahul Sharma	731
12	Riya Mehta	731
13	Rishabh Sharma	731
14	Vishwan Kumar	731
- Message Bar:** Displays "Total rows: 97 of 97" and "Query complete 00:00:00.085".
- Status Bar:** Shows "Ln 1946, Col 10" and "5:25 PM".

Number of tuples: 97

20) Show employers offering multiple types.

```
SELECT Employer_Name, Opportunity_Type,  
COUNT(Opportunity_Type)  
FROM Employer  
JOIN Opportunities ON Employer.Employer_ID =  
Opportunities.Employer_ID  
GROUP BY Employer_Name, Opportunity_Type  
HAVING COUNT(Opportunity_Type) > 1;
```

The screenshot shows the pgAdmin 4 interface with a query editor window. The title bar says "Youngsters_db/postgres@PostgreSQL 16". The query tab contains the following SQL code:

```
--20 Show employers offering multiple types.  
SELECT Employer_Name, Opportunity_Type, COUNT(Opportunity_Type)  
FROM Employer  
JOIN Opportunities ON Employer.Employer_ID = Opportunities.Employer_ID  
GROUP BY Employer_Name, Opportunity_Type  
HAVING COUNT(Opportunity_Type) > 1;
```

The results tab displays a table with the following data:

	employer_name	opportunity_type	count
1	ITC Hotels	Internship	2
2	Nykaa	Full-time	2
3	Air India	Freelance	2
4	L&T	Freelance	2
5	Razorpay	Freelance	2
6	Nestle	Contract	2
7	PharmEasy	Freelance	2
8	Oracle	Part-time	2
9	Reliance Jio	Contract	2
10	NTPC	Internship	2
11	Zomato Gold	Internship	2
12	Bharat Forge	Internship	2
13	BSNL	Freelance	2
14	Starbucks	Internship	2

Total rows: 18 of 18 Query complete 00:00:00.060 Ln 1953, Col 36

Number of tuples: 18

21 Display only current employment statuses.

```
CREATE VIEW Active_Employment AS
SELECT Name, Industry_Type, Start_Date, End_Date
FROM Youngster
JOIN Employment ON Youngster.Employment_ID =
Employment.JOB_ID
WHERE End_Date IS NULL OR End_Date > CURRENT_DATE;

SELECT* FROM Active_Employment;
```

The screenshot shows a PostgreSQL client interface with a query editor and a results table. The query editor contains the SQL code from the previous section. The results table displays 72 tuples of current employment statuses.

	name	industry_type	start_date	end_date
1	Aarav Sharma	Information Technology	2022-01-01	2025-01-01
2	Ananya Gupta	Healthcare	2021-02-01	2024-12-01
3	Vihaan Kumar	Finance	2023-03-01	2025-03-01
4	Aanya Verma	Education	2022-04-01	[null]
5	Pooja Singh	Manufacturing	2019-06-01	[null]
6	Kabir Khan	Retail	2023-07-01	2025-07-01
7	Rohan Patel	Transportation	2022-09-01	[null]
8	Sal Kumar	Real Estate	2022-11-01	2025-11-01
9	Nisha Yadav	Legal	2020-12-01	[null]
10	Ishaan Joshi	Marketing	2023-01-01	2024-12-01
11	Sneha Desai	Media	2021-02-01	[null]
12	Tara Rani	Information Security	2023-04-01	2025-04-01

Number of tuples: 72

22) List cities where no health facilities are available.

```
SELECT City.City_ID,City.City_Name  
FROM City  
LEFT JOIN Health_Facility ON City.City_ID =  
Health_Facility.Facility_ID  
WHERE Health_Facility.Facility_ID IS NULL;
```

The screenshot shows a PostgreSQL client interface with the following details:

- Query History:** Displays the SQL query executed, which lists cities where no health facilities are available. The query includes a comment, a SELECT statement, and a WHERE clause filtering for null Facility_ID.
- Data Output:** Shows the results of the query as a table. The table has two columns: `city_id` and `city_name`. The data consists of 41 rows, each representing a city with its ID and name.
- Table Data:**

	city_id	city_name
1	101	Darbhanga
2	102	Gandhinagar
3	103	Kharagpur
4	104	Bangalore
5	105	Tiruchirappalli
6	106	Surathkal
7	107	Vellore
8	108	Manipal
9	109	Phagwara
10	110	Thanjavur
11	111	Mesra
12	113	Sri City
13	115	Neemrana
14	116	Pune
15	117	Gharuan
- System Status:** The bottom right corner shows system status icons for battery, signal, and date/time (5:37 PM, 11/2/2024).

Number of tuples: 41

23) Find the average salary offered across all jobs within each state.

```
SELECT State.State_Name, AVG(Employment.Salary) AS Avg_Salary
FROM Employment
JOIN Youngster ON Employment.JOB_ID = Youngster.Employment_ID
JOIN Education ON Youngster.Education_ID =
Education.Education_ID
JOIN Institute ON Education.Institute_ID = Institute.Institution_ID
JOIN City ON Institute.City_ID = City.City_ID
JOIN State ON City.State_ID = State.State_ID
GROUP BY State.State_Name;
```

The screenshot shows a PostgreSQL client interface with the following details:

- Query Editor:** The query is pasted into the editor, starting with a comment to skip the first part of the question (lines 1967-1970). It then selects the average salary for each state from the State table, joining through Employment, Youngster, Education, Institute, City, and State tables.
- Data Output:** The results are displayed in a table with two columns: state_name and avg_salary. The data shows 19 rows, corresponding to the 19 states listed in the output.

state_name	avg_salary
Uttarakhand	50000.00000000000000
Rajasthan	77500.00000000000000
Jharkhand	61250.00000000000000
Gujarat	63500.00000000000000
Himachal Pradesh	35000.00000000000000
Karnataka	58888.88888888888889
Kerala	62000.00000000000000
Delhi	66333.33333333333333
Assam	47500.00000000000000
... (16 more rows)	...

Total rows: 19 of 19 Query complete 00:00:00.078 Ln 1971, Col 73

Number of tuples: 19

24) Calculate the sum of credits each youngster has earned.

```
SELECT Youngster.Name, SUM(Credits) AS Total_Credits  
FROM Youngster  
JOIN Enrolled ON Youngster.Youngster_ID = Enrolled.Youngster_ID  
GROUP BY Youngster.Name;
```

The screenshot shows a PostgreSQL query editor window. The top part displays the SQL query:

```
1981 --24 Calculate the sum of credits each youngster has earned.  
1982 ✓ SELECT Youngster.Name, SUM(Credits) AS Total_Credits  
1983   FROM Youngster  
1984   JOIN Enrolled ON Youngster.Youngster_ID = Enrolled.Youngster_ID  
1985   GROUP BY Youngster.Name;  
1986
```

The bottom part shows the resulting data table:

	name	total_credits
1	Akash Bansal	29
2	Harish Rao	35
3	Sidharth Kumar	40
4	Rohan Patel	30
5	Deepika Agarwal	30
6	Karan Singh	21
7	Parul Yadav	31
8	Sonali Roy	28
9	Harsh Mehta	32
10	Raj Kumar	32
11	Sonal Gupta	27
12	Manoj Kumar	37
13	Riya Sharma	37
14	Shweta Mehta	34
15	Preeti Chawla	33

Total rows: 97 Query complete 00:00:00.064 Ln 1986, Col 1

Number of tuples: 97

25) Show the institutes with top tuition 10 fees.

```
SELECT Name, Tuition_Fees  
FROM Institute  
ORDER BY Tuition_Fees DESC  
LIMIT 10;
```

The screenshot shows the pgAdmin 4 interface with a query editor window. The query is:

```
--25 Show the institutes with top 10 tuition fees.  
SELECT Name, Tuition_Fees  
FROM Institute  
ORDER BY Tuition_Fees DESC  
LIMIT 10;
```

The results are displayed in a table:

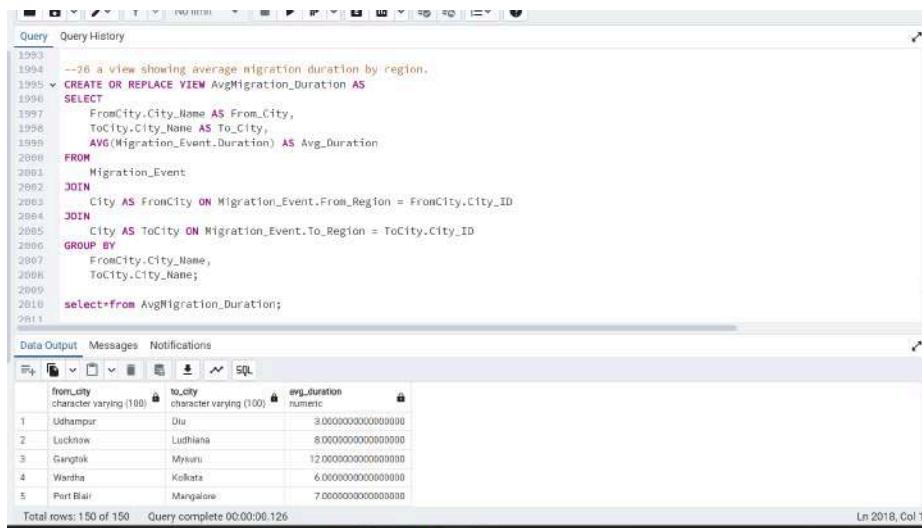
	name	tuition_fees
1	IIM Calcutta	250000.00
2	Indian School of Business	250000.00
3	Indian Institute of Management, Lucknow	250000.00
4	Indian Institute of Management, Kozhikode	250000.00
5	Indian Institute of Management, Ranchi	250000.00
6	IIM Bangalore	250000.00
7	IIM Ahmedabad	250000.00
8	Indian Institute of Technology, Dhanbad	200000.00
9	IIT Kharagpur	200000.00
10	Vellore Institute of Technology	200000.00

Total rows: 10 of 10 Query complete 00:00:00.065 Ln 1988, Col 37

Number of tuples: 10

26) a view showing average migration duration by region.

```
CREATE OR REPLACE VIEW AvgMigration_Duration AS
SELECT
    FromCity.City_Name AS From_City,
    ToCity.City_Name AS To_City,
    AVG(Migration_Event.Duration) AS Avg_Duration
FROM
    Migration_Event
JOIN
    City AS FromCity ON Migration_Event.From_Region =
FromCity.City_ID
JOIN
    City AS ToCity ON Migration_Event.To_Region =
ToCity.City_ID
GROUP BY
    FromCity.City_Name,
    ToCity.City_Name;
select*from AvgMigration_Duration;
```



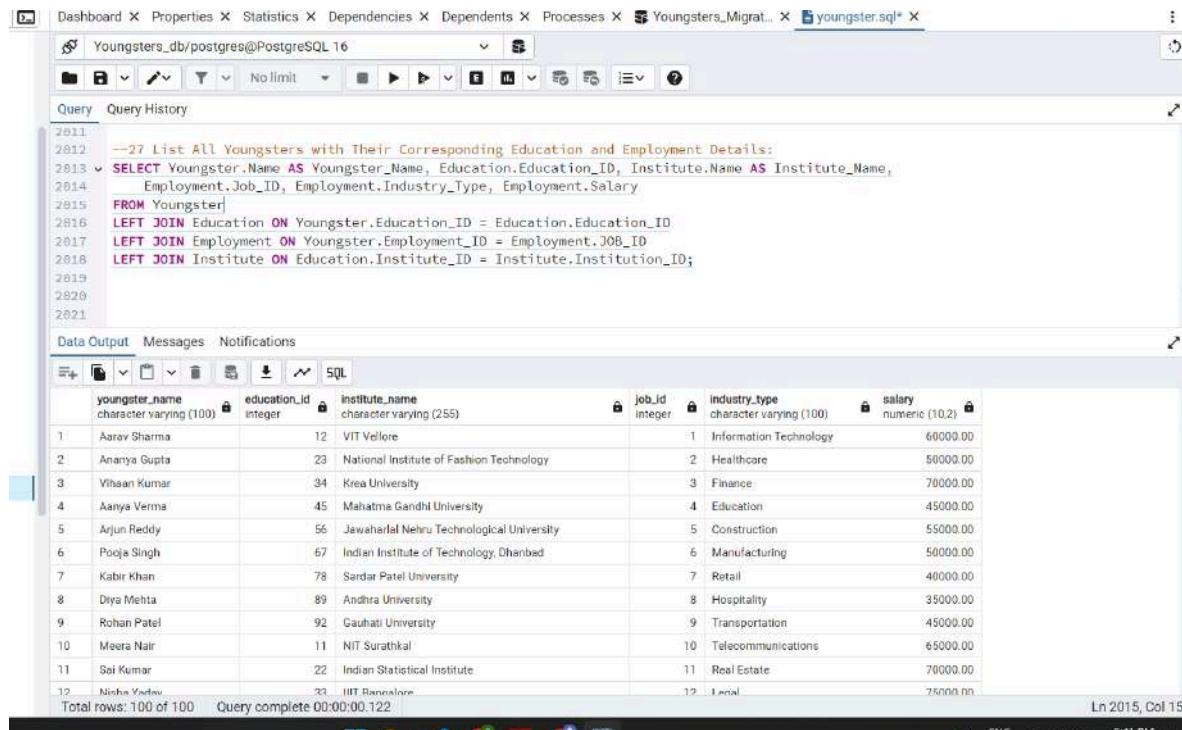
The screenshot shows a database interface with two panes. The top pane is the 'Query' pane, displaying the SQL code for creating a view named 'AvgMigration_Duration'. The bottom pane is the 'Data Output' pane, showing the results of the query. The results are a table with three columns: 'from_city', 'to_city', and 'avg_duration'. The data consists of five rows, each representing a migration event between two cities. The 'avg_duration' column contains large numerical values representing time in microseconds.

	from_city	to_city	avg_duration
1	Udhampur	Diu	3.000000000000000
2	Lucknow	Ludhiana	8.000000000000000
3	Gangtok	Mykun	12.000000000000000
4	Wardha	Kolkata	6.000000000000000
5	Port Blair	Mangalore	7.000000000000000

Number of tuples: 150

27) List All Youngsters with Their Corresponding Education and Employment Details:

```
SELECT Youngster.Name AS Youngster_Name,
       Education.Education_ID, Institute.Name AS Institute_Name,
       Employment.Job_ID, Employment.Industry_Type,
       Employment.Salary
  FROM Youngster
 LEFT JOIN Education ON Youngster.Education_ID =
       Education.Education_ID
 LEFT JOIN Employment ON Youngster.Employment_ID =
       Employment.JOB_ID
 LEFT JOIN Institute ON Education.Institute_ID =
       Institute.Institution_ID;
```



The screenshot shows the pgAdmin 4 interface with the following details:

- Toolbar:** Dashboard, Properties, Statistics, Dependencies, Dependents, Processes, Youngsters_Migrat..., youngster.sql*
- Query Editor:** Shows the SQL code from the previous section.
- Data Output:** A table displaying the results of the query. The columns are:
 - youngster_name (character varying(100))
 - education_id (integer)
 - institute_name (character varying(255))
 - Job_id (integer)
 - industry_type (character varying(100))
 - salary (numeric(10,2))The data consists of 100 rows, each representing a youngster with their education and employment details. The last row is partially visible.
- Bottom Status Bar:** Total rows: 100 of 100, Query complete 00:00:00.122, Ln 2015, Col 15.

Number of tuples: 100

28) Count the Number of Youngsters from Each State

```
SELECT State.State_Name, COUNT(Youngster.Youngster_ID)
AS Youngster_Count
FROM Youngster
JOIN City ON Youngster.Place_of_Origin = City.City_Name
JOIN State ON City.State_ID = State.State_ID
GROUP BY State.State_Name;
```

The screenshot shows the pgAdmin 4 interface with the following details:

- Object Explorer:** Shows the database structure with the "Tables" node selected.
- Query Editor:** Contains the SQL code for the query.
- Data Output:** Displays the results of the query in a table format.

SQL Query:

```
--28 Count the Number of Youngsters from Each State
SELECT State.State_Name, COUNT(Youngster.Youngster_ID) AS Youngster_Count
FROM Youngster
JOIN City ON Youngster.Place_of_Origin = City.City_Name
JOIN State ON City.State_ID = State.State_ID
GROUP BY State.State_Name;
```

Result Table:

state_name	youngster_count
Uttarakhand	1
Rajasthan	7
Gujarat	8
Karnataka	6
Kerala	2
Delhi	8
Assam	1
West Bengal	4
Andhra Pradesh	1
Madhya Pradesh	7
Chhattisgarh	3
Jammu and Kashmir	1
Odisha	1
Maharashtra	16
Bihar	1

Total rows: 20 of 20 Query complete 00:00:00.058

Number of tuples: 20

29) Count the Number of Health Facilities by Facility Type

```
SELECT Facility_Type, COUNT(Facility_ID) AS Facility_Count  
FROM Health_Facility  
GROUP BY Facility_Type;
```

The screenshot shows the pgAdmin interface with a query editor window. The query is:

```
--29 Count the Number of Health Facilities by Facility Type  
SELECT Facility_Type, COUNT(Facility_ID) AS Facility_Count  
FROM Health_Facility  
GROUP BY Facility_Type;
```

The results are displayed in a table:

	facility_type	facility_count
1	Hospital	9
2	Psychiatric Hospital	1
3	Women's Hospital	3
4	Diagnostics Center	1
5	Surgical Hospital	1
6	Heart Hospital	2
7	Nursing Home	2
8	Multi-Specialty Hospital	75
9	Clinic	1
10	Eye Hospital	1
11	Cancer Hospital	4

Total rows: 11 of 11 Query complete 00:00:00.066 Ln 2029, Col 21

Number of tuples: 11

30) Average Tuition Fees by City

```
SELECT City.City_Name, AVG(Institute.Tuition_Fees) AS  
Avg_Tuition_Fees  
FROM Institute  
JOIN City ON Institute.City_ID = City.City_ID  
GROUP BY City.City_Name;
```

The screenshot shows a PostgreSQL client interface with the following details:

- Toolbar:** Includes icons for Dashboard, Properties, Statistics, Dependencies, Dependents, Processes, and two tabs labeled "Youngsters_Migrat..." and "youngster.sql*".
- Query Editor:** Shows the SQL query from the question above.
- Data Output:** Displays the results of the query in a table format.
- Table Data:**

	city_name	avg_tuition_fees
1	Jaipur	70000.000000000000
2	Faridkot	70000.000000000000
3	Kharagpur	200000.000000000000
4	Kochi	70000.000000000000
5	Mandi	200000.000000000000
6	Kottayam	50000.000000000000
7	Wardha	60000.000000000000
8	Dhanbad	200000.000000000000
9	Vijayawada	80000.000000000000
10	Warangal	50000.000000000000
11	Bhilai	200000.000000000000
12	Delhi	77692.307692307692
13	Pilani	150000.000000000000
14	Bhopal	60000.000000000000
15	Maninagar	120000.000000000000

Total rows: 56 of 56 Query complete 00:00:00.137 Ln 2033, Col 34

Number of tuples: 56

31) Get the Total Count of Migration Events Grouped by From_Region

```
SELECT City.City_ID AS From_Region_City_ID,
City.City_Name AS From_Region_City_Name,
        COUNT(Migration_Event.Migration_ID) AS
Total_Migrations
FROM Migration_Event
JOIN City ON Migration_Event.From_Region = City.City_ID
GROUP BY City.City_ID, City.City_Name
ORDER BY City_ID;
```

The screenshot shows the pgAdmin 4 interface with a query editor and a results grid.

Query Editor:

```
--31 Get the Total Count of Migration Events Grouped by From_Region
SELECT City.City_ID AS From_Region_City_ID, City.City_Name AS From_Region_City_Name,
        COUNT(Migration_Event.Migration_ID) AS Total_Migrations
FROM Migration_Event
JOIN City ON Migration_Event.From_Region = City.City_ID
GROUP BY City.City_ID, city.city_name
ORDER BY City_ID;
```

Results Grid:

from_region_city_id	from_region_city_name	total_migrations
1	Vijayawada	1
2	Itanagar	1
3	Guwahati	1
4	Patna	2
5	Raipur	1
6	Panaji	1
7	Ahmedabad	1
8	Gurugram	1
9	Shimla	2
10	Ranchi	1
11	Bengaluru	1
12	Thiruvananthapuram	1
13	Bhopal	1

Total rows: 136 of 136 Query complete 00:00:00.066 Ln 2048, Col 1

Number of tuples: 136

32) Government Policies Affecting Regions

```
SELECT State.State_Name,  
COUNT(Government_Policy.Policy_ID) AS Policy_Count  
FROM Government_Policy  
JOIN State ON Government_Policy.State_ID = State.State_ID  
GROUP BY State.State_Name  
ORDER BY COUNT(Government_Policy.Policy_ID) DESC;
```

The screenshot shows the pgAdmin 4 interface with a query editor and a results grid.

Query Editor:

```
--32 Government Policies Affecting Regions  
SELECT State.State_Name, COUNT(Government_Policy.Policy_ID) AS Policy_Count  
FROM Government_Policy  
JOIN State ON Government_Policy.State_ID = State.State_ID  
GROUP BY State.State_Name  
ORDER BY COUNT(Government_Policy.Policy_ID) DESC;
```

Results Grid:

	state_name	policy_count
1	Kerala	5
2	Punjab	5
3	Tamil Nadu	5
4	West Bengal	4
5	Bihar	4
6	Assam	4
7	Haryana	4
8	Goa	4
9	Odisha	4
10	Gujarat	4
11	Himachal Pradesh	4

Total rows: 36 of 36 Query complete 00:00:00.060 Ln 2052, Col 49

Number of tuples: 36

33) Migration Patterns Over Time

```
SELECT EXTRACT(YEAR FROM Migration_Date) AS  
Migration_Year, COUNT(Migration_ID) AS Total_Migrations  
FROM Migration_Event  
GROUP BY Migration_Year  
ORDER BY Migration_Year;
```

The screenshot shows the pgAdmin 4 interface with a query editor and a results grid.

Query Editor:

```
--33 Migration Patterns Over Time  
SELECT EXTRACT(YEAR FROM Migration_Date) AS Migration_Year, COUNT(Migration_ID) AS Total_Migrations  
FROM Migration_Event  
GROUP BY Migration_Year  
ORDER BY Migration_Year;
```

Data Output Grid:

	migration_year	total_migrations
1	2014	6
2	2015	15
3	2016	13
4	2017	10
5	2018	15
6	2019	21
7	2020	18
8	2021	17
9	2022	19
10	2023	14
11	2024	2

Total rows: 11 of 11 Query complete 00:00:00.057 Ln 2055, Col 34

Number of tuples: 11

34) Youngsters with multiple enrollments

```
SELECT Youngster.Name, COUNT(Enrolled.Institute_ID) AS Enrollment_Count
FROM Youngster
JOIN Enrolled ON Youngster.Youngster_ID = Enrolled.Youngster_ID
GROUP BY Youngster.Name
HAVING COUNT(Enrolled.Institute_ID) > 1;
```

The screenshot shows a PostgreSQL query editor window. The title bar indicates the connection is to 'Youngsters_db/postgres@PostgreSQL 16'. The main area contains the SQL query:

```
--34 Youngsters with multiple enrollments
SELECT Youngster.Name, COUNT(Enrolled.Institute_ID) AS Enrollment_Count
FROM Youngster
JOIN Enrolled ON Youngster.Youngster_ID = Enrolled.Youngster_ID
GROUP BY Youngster.Name
HAVING COUNT(Enrolled.Institute_ID) > 1;
```

Below the query, the results are displayed in a table:

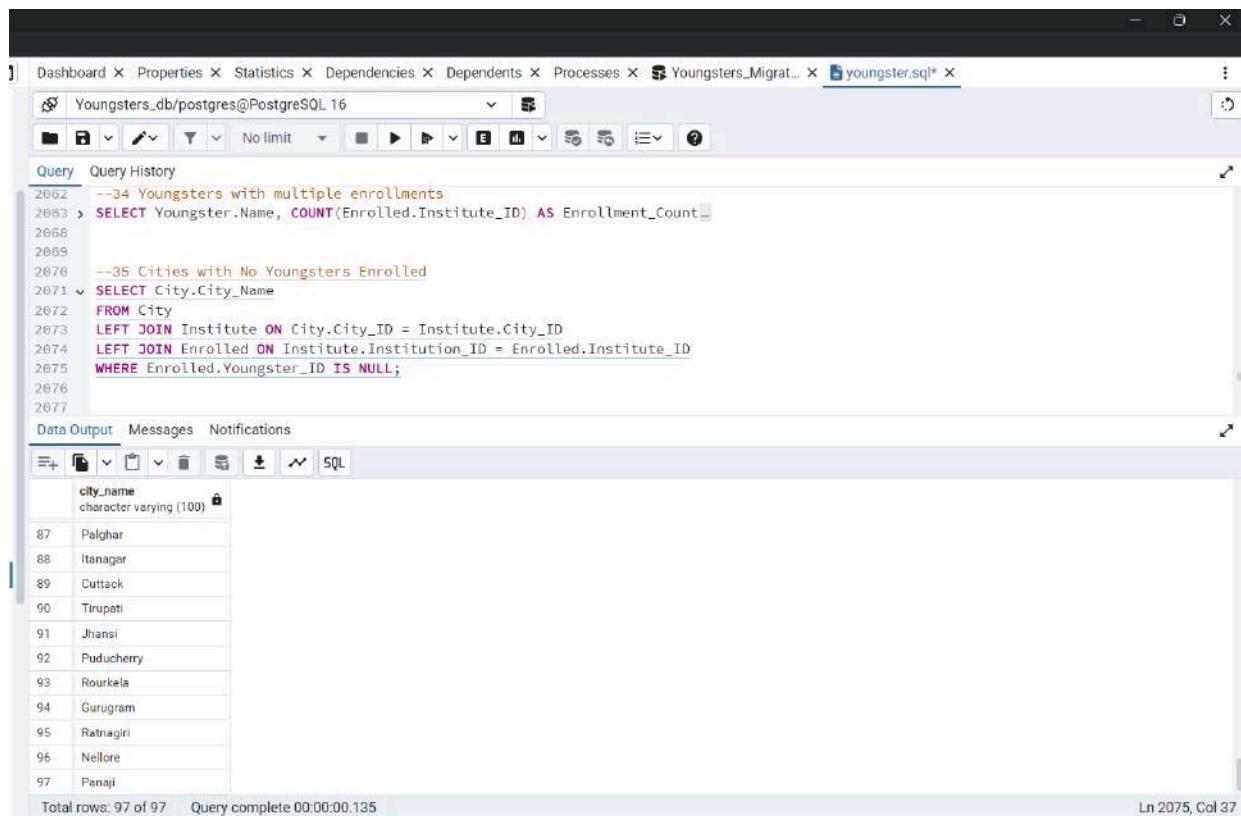
	name	enrollment_count
1	Rahul Sharma	2
2	Sakshi Singh	2
3	Ravi Kumar	2

At the bottom of the window, status information is shown: 'Total rows: 3 of 3' and 'Query complete 00:00:00.068'. The bottom right corner shows the system time as '7:11 PM'.

Number of tuples: 3

35) Cities with No Youngsters Enrolled

```
SELECT City.City_Name
FROM City
LEFT JOIN Institute ON City.City_ID = Institute.City_ID
LEFT JOIN Enrolled ON Institute.Institution_ID =
Enrolled.Institute_ID
WHERE Enrolled.Youngster_ID IS NULL;
```



The screenshot shows the pgAdmin 4 interface with a query editor window. The title bar says "Youngsters_db/postgres@PostgreSQL 16". The query history pane contains two entries:

- 34 Youngsters with multiple enrollments
- 35 Cities with No Youngsters Enrolled

The second entry is expanded, showing the full SQL code:

```
2662 --34 Youngsters with multiple enrollments
2663 > SELECT Youngster.Name, COUNT(Enrolled.Institute_ID) AS Enrollment_Count
2664
2665
2670 --35 Cities with No Youngsters Enrolled
2671 < SELECT City.City_Name
2672 FROM City
2673 LEFT JOIN Institute ON City.City_ID = Institute.City_ID
2674 LEFT JOIN Enrolled ON Institute.Institution_ID = Enrolled.Institute_ID
2675 WHERE Enrolled.Youngster_ID IS NULL;
2676
2677
```

The results pane displays a table with one column "city_name" containing 97 rows:

city_name
Palghar
Itanagar
Cuttack
Tirupati
Jhansi
Puducherry
Rourkela
Gurugram
Ratnagiri
Nellore
Panaji

Total rows: 97 of 97 Query complete 00:00:00.135 Ln 2075, Col 37

Number of tuples: 97

36) Top 5 States with Most Migration Events

```
SELECT State.State_Name,  
COUNT(Migration_Event.Migration_ID) AS Total_Migrations  
FROM Migration_Event  
JOIN City ON Migration_Event.From_Region = City.City_ID  
JOIN State ON City.State_ID = State.State_ID  
GROUP BY State.State_Name  
ORDER BY Total_Migrations DESC  
LIMIT 5;
```

The screenshot shows a PostgreSQL client interface with the following details:

- Connection:** Youngsters_db/postgres@PostgreSQL 16
- Query History:** The query is listed in the history.
- Result Table:** A table titled "SQL" displays the top 5 states with their total migrations.

	state_name	total_migrations
1	Maharashtra	19
2	Rajasthan	11
3	Uttar Pradesh	9
4	Gujarat	8
5	Kerala	8
- Statistics:** Total rows: 5 of 5. Query complete 00:00:00.066. Line 2085, Col 1.

Number of tuples: 5

37) the average salary of youngsters segmented by their age groups (under 18, 18-24, 25-30, and over 30)

SELECT CASE

WHEN EXTRACT(YEAR FROM AGE(CURRENT_DATE,
Youngster.Date_Of_Birth)) < 18 THEN 'Under 18'

WHEN EXTRACT(YEAR FROM AGE(CURRENT_DATE,
Youngster.Date_Of_Birth)) BETWEEN 18 AND 24 THEN '18-24'

WHEN EXTRACT(YEAR FROM AGE(CURRENT_DATE,
Youngster.Date_Of_Birth)) BETWEEN 25 AND 30 THEN '25-30'

ELSE 'Over 30'

END AS Age_Group, AVG(Employment.Salary) AS
Avg_Salary

FROM Youngster

JOIN Employment ON Youngster.Employment_ID =
Employment.JOB_ID
GROUP BY Age_Group
ORDER BY Age_Group;

```
--37) the average salary of youngsters segmented by their age groups (under 18, 18-24, 25-30, and over 30)
2087
2088 v  SELECT CASE
2089   WHEN EXTRACT(YEAR FROM AGE(CURRENT_DATE, Youngster.Date_Of_Birth)) < 18 THEN 'Under 18'
2090   WHEN EXTRACT(YEAR FROM AGE(CURRENT_DATE, Youngster.Date_Of_Birth)) BETWEEN 18 AND 24 THEN '18-24'
2091   WHEN EXTRACT(YEAR FROM AGE(CURRENT_DATE, Youngster.Date_Of_Birth)) BETWEEN 25 AND 30 THEN '25-30'
2092   ELSE 'Over 30'
2093 END AS Age_Group, AVG(Employment.Salary) AS Avg_Salary
2094 FROM Youngster
2095 JOIN Employment ON Youngster.Employment_ID = Employment.JOB_ID
2096 GROUP BY Age_Group
2097 ORDER BY Age_Group;
2098
2099
2100
2101
```

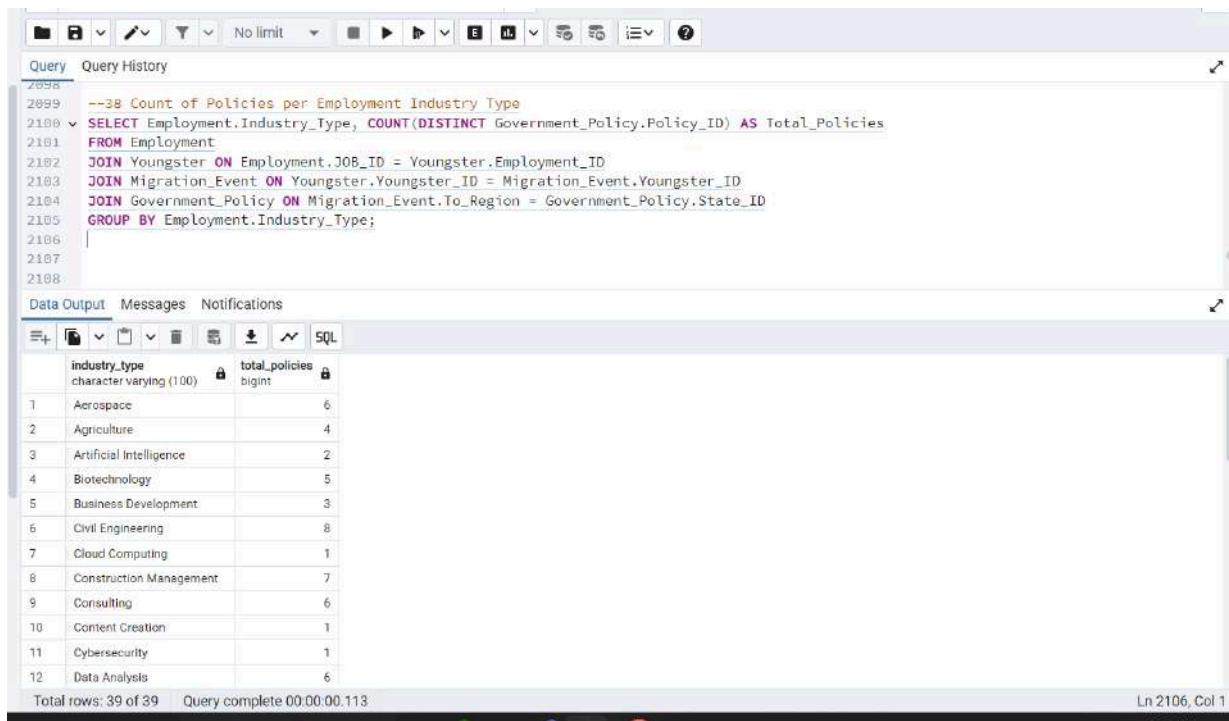
age_group	avg_salary
18-24	64062.580000000000
25-30	64047.619047619048

Total rows: 2 of 2 Query complete 00:00:00.055 Ln 2094, Col 15

Number of tuples: 2

38) Count of Policies per Employment Industry Type

```
SELECT Employment.Industry_Type, COUNT(DISTINCT
Government_Policy.Policy_ID) AS Total_Policies
FROM Employment
JOIN Youngster ON Employment.JOB_ID = Youngster.Employment_ID
JOIN Migration_Event ON Youngster.Youngster_ID =
Migration_Event.Youngster_ID
JOIN Government_Policy ON Migration_Event.To_Region =
Government_Policy.State_ID
GROUP BY Employment.Industry_Type;
```



The screenshot shows a database query interface with the following details:

- Query History:** Displays the executed SQL code, which is identical to the one provided in the text above.
- Data Output:** Shows the results of the query in a table format. The table has two columns: `industry_type` (character varying (100)) and `total_policies` (bigint). The data is as follows:

industry_type	total_policies
Aerospace	6
Agriculture	4
Artificial Intelligence	2
Biotechnology	5
Business Development	3
Civil Engineering	8
Cloud Computing	1
Construction Management	7
Consulting	6
Content Creation	1
Cybersecurity	1
Data Analysis	6

- Message Bar:** Shows "Total rows: 39 of 39" and "Query complete 00:00:00.113".
- Status Bar:** Shows "Ln 2106, Col 1".

Number of tuples: 39

39) Average Salary by Policy Type

```
SELECT Government_Policy.Policy_Type, AVG(Employment.Salary)
AS Avg_Salary
FROM Employment
JOIN Youngster ON Employment.JOB_ID = Youngster.Employment_ID
JOIN Migration_Event ON Youngster.Youngster_ID =
Migration_Event.Youngster_ID
JOIN Government_Policy ON Migration_Event.To_Region =
Government_Policy.State_ID
GROUP BY Government_Policy.Policy_Type;
```

The screenshot shows a PostgreSQL client interface with the following details:

- Query History:** Displays the SQL code for question 39.
- Data Output:** Shows the results of the query in a table format.
- Table Data:**

	policy_type	avg_salary
1	Job	60641.025641025641
2	Migration	61388.888888888889
3	Education	59375.000000000000
4	Health	59285.714285714286
- Status Bar:** Shows "Total rows: 4 of 4" and "Query complete 00:00:00.079".
- System Tray:** Shows system icons and the date/time "7:38 PM".

Number of tuples: 4

40) Salary Trends Based on Government Policies

```
SELECT Government_Policy.Policy_Name, AVG(Employment.Salary)
AS Avg_Salary
FROM Employment
JOIN Youngster ON Employment.JOB_ID = Youngster.Employment_ID
JOIN Migration_Event ON Youngster.Youngster_ID =
Migration_Event.Youngster_ID
JOIN Government_Policy ON Migration_Event.To_Region =
Government_Policy.State_ID
GROUP BY Government_Policy.Policy_Name
ORDER BY Avg_Salary DESC;
```

The screenshot shows the pgAdmin interface with a query editor and a data output viewer.

Query Editor:

```
2114
2115 --40 Salary Trends Based on Government Policies
2116 SELECT Government_Policy.Policy_Name, AVG(Employment.Salary) AS Avg_Salary
2117 FROM Employment
2118 JOIN Youngster ON Employment.JOB_ID = Youngster.Employment_ID
2119 JOIN Migration_Event ON Youngster.Youngster_ID = Migration_Event.Youngster_ID
2120 JOIN Government_Policy ON Migration_Event.To_Region = Government_Policy.State_ID
2121 GROUP BY Government_Policy.Policy_Name
2122 ORDER BY Avg_Salary DESC;
2123
```

Data Output:

	policy_name	avg_salary
1	Delhi Migrant Welfare Act	95000.000000000000
2	Delhi Migration Support Policy	95000.000000000000
3	Dadra and Nagar Haveli Job Opportunity Scheme	90000.000000000000
4	Assam Job Guarantee Act	85000.000000000000
5	Assam Scholarship Program	85000.000000000000
6	Assam Health Protection Scheme	85000.000000000000
7	Interstate Migrant Workers Act	85000.000000000000
8	Kerala Student Welfare Scheme	80000.000000000000
9	Kerala Startup Job Program	80000.000000000000
10	Andaman and Nicobar Health Program	80000.000000000000
11	Kerala Migrant Welfare Program	80000.000000000000
12	Kerala Healthcare Assistance	80000.000000000000

Total rows: 99 of 99 Query complete 00:00:00.063 Ln 2123, Col 1

Number of tuples: 99

Chapter 5: Interface Implementation

5.1 Setup JDBC and Basic GUI

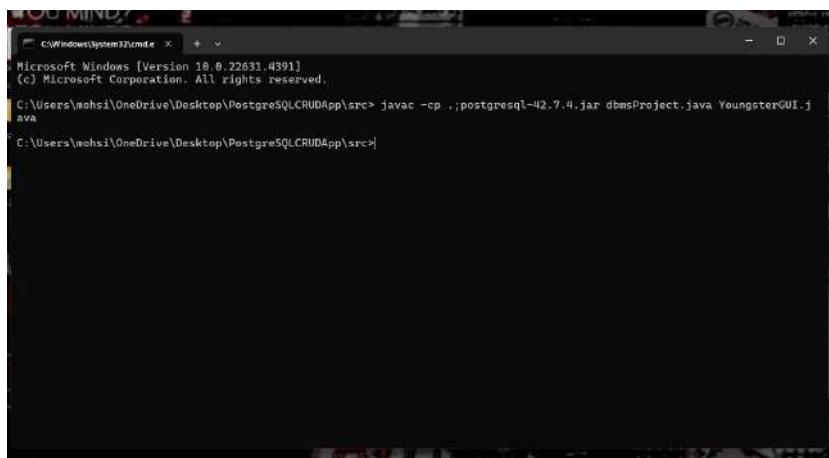
Setup Steps for JDK, postgres, and JDBC Driver

- 1) Ensure JDK and PostgreSQL are Installed: You need Java Development Kit (JDK) and PostgreSQL installed.
- 2) Download JDBC Driver: Download the PostgreSQL JDBC driver (postgresql-42.7.4.jar) from <https://jdbc.postgresql.org/download/>.

Steps to create the GUI project folder

- 1) Create a New Project Directory:
 - Create a directory for your project, e.g., PostgreSQLCRUDApp.
- 2) Create Subdirectory for Your Code:
 - Inside your project folder, create a src folder for your Java code.
- 3) Place the JDBC Driver:
 - Put the postgresql-42.7.4.jar file inside your project folder (src).

Compilation Process



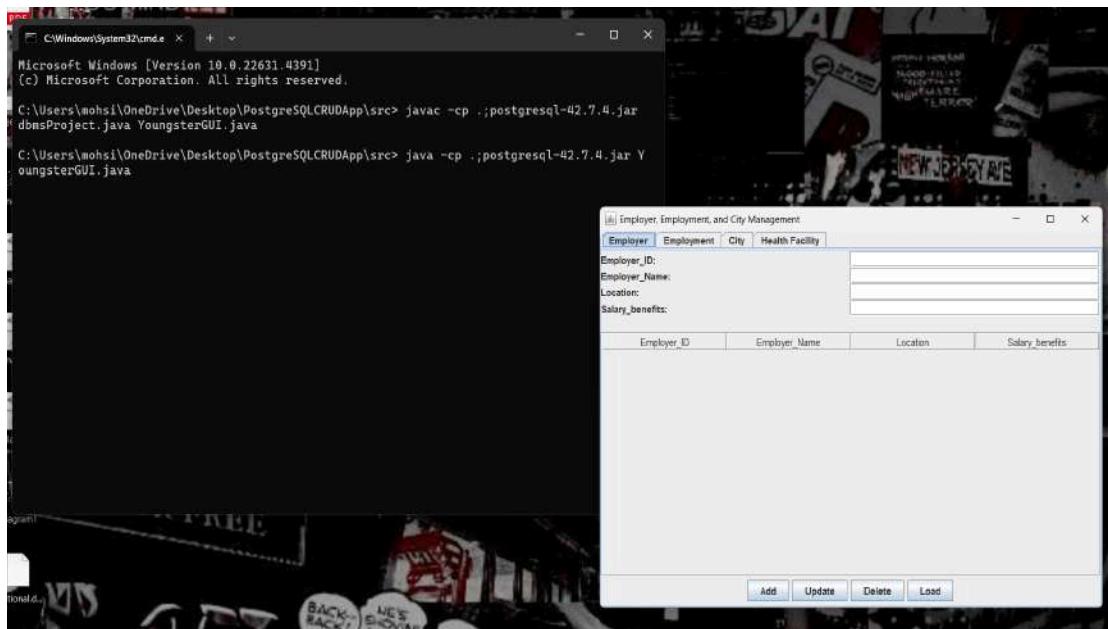
A screenshot of a Windows Command Prompt window titled 'CMD MINDY'. The window shows the following command being run:

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.22631.4391]
(c) Microsoft Corporation. All rights reserved.

C:\Users\mohsi\OneDrive\Desktop\PostgreSQLCRUDApp\src> javac -cp .;postgresql-42.7.4.jar dbmsProject.java YoungsterGUI.java

C:\Users\mohsi\OneDrive\Desktop\PostgreSQLCRUDApp\src>
```

Running the GUI



The screenshot shows the 'Employer' tab of the Java application window. The table displays the following data:

Employer_ID	Employer_Name	Location	Salary_benefits
1	Tata Consultancy Services	Mumbai	80000
2	Infosys	Bengaluru	75000
3	Wipro	Bengaluru	70000
4	HCL Technologies	Kerala	72000
5	Accenture	Gurgaon	90000
6	Capgemini	Chennai	85000
7	Tech Mahindra	Pune	60000
8	Cognizant	Mumbai	75000
9	IBM	Bengaluru	85000
10	Oracle	Hyderabad	90000
11	SAP	Gurgaon	85000
12	Dell	Bengaluru	78000
13	Intel	Hyderabad	93000
14	Microsoft	Hyderabad	100000
15	Amazon	Bengaluru	120000
16	Google	Hyderabad	110000
17	Facebook	Hyderabad	115000
18	Adobe	Kerala	85000
19	Salesforce	Bengaluru	80000
20	Accenture	Bengaluru	68000
21	Uber	Bengaluru	100000
22	Ola	Bengaluru	70000
23	Zomato	Gurgaon	60000
24	Swiggy	Bengaluru	65000
25	Paytm	Kerala	75000
26	Razorpay	Bengaluru	70000
27	PhonePe	Bengaluru	72000
28	Mynt	Bengaluru	68000
29	Nykaa	Mumbai	60000
30	Flipkart	Bengaluru	95000
31	Shapetal	Gurgaon	90000
32	Tata Steel	Jharkhand	70000
33	Axence Industries	Mumbai	65000
34	Hindustan Unilever	Mumbai	60000
35	ITC	Kolkata	65000
36	I.A.T	Mumbai	80000
37	Godrej	Mumbai	75000
38	Mahindra & Mahindra	Mumbai	80000

Employee Employment and City Management					
Employer	Employment	City	Health Facility		
Job_ID	Industry_Type	Salary	Start_Date	End_Date	
1	Information Technology	60000	2022-01-01	2025-01-01	
2	Healthcare	50000	2021-02-01	2024-12-01	
3	Finance	70000	2023-03-01	2025-03-01	
4	Education	45000	2022-04-01	2024-04-01	
5	Construction	55000	2020-05-01	2023-06-01	
6	Manufacturing	50000	2019-06-01	2024-10-01	
7	Retail	40000	2023-07-01	2025-07-01	
8	Hospitality	35000	2020-08-01	2023-08-01	
9	Transportation	45000	2022-09-01	2024-09-01	
10	Telecommunications	65000	2021-10-01	2024-10-01	
11	Real Estate	70000	2022-11-01	2025-11-01	
12	Automotive	75000	2023-12-01	2025-12-01	
13	Marketing	40000	2023-01-01	2024-12-01	
14	Media	40000	2021-02-01	2023-03-01	
15	Agriculture	30000	2020-03-01	2023-03-01	
16	Information Security	90000	2023-04-01	2025-04-01	
17	Pharmaceuticals	90000	2021-05-01	2024-05-01	
18	Aerospace	75000	2022-06-01	2024-06-01	
19	Biotechnology	85000	2023-07-01	2025-07-01	
20	Consulting	65000	2022-08-01	2024-08-01	
21	Entertainment	65000	2023-09-01	2025-09-01	
22	Insurance	60000	2021-10-01	2024-10-01	
23	Textiles	40000	2022-11-01	2024-11-01	
24	Food Service	35000	2021-12-01	2023-12-01	
25	Construction Management	60000	2023-01-01	2025-01-01	
26	Civil Engineering	70000	2022-02-01	2024-02-01	
27	Environmental Science	50000	2023-03-01	2024-03-01	
28	Data Analysis	40000	2023-04-01	2024-04-01	
29	Web Development	60000	2023-05-01	2024-05-01	
30	Graphic Design	40000	2023-06-01	2024-06-01	
31	Interior Design	70000	2021-07-01	2024-07-01	
32	Content Creation	30000	2022-08-01	2024-08-01	
33	Event Planning	35000	2020-09-01	2023-09-01	
34	Sales	45000	2021-10-01	2024-10-01	
35	Human Resources	50000	2023-01-01	2025-01-01	
36	Supply Chain Management	60000	2021-02-01	2023-02-01	
37	Project Portfolio	55000	2022-03-01	2024-01-01	

Employer, Employment, and City Management			
Employer	Employment	City	Health Facility
City_ID:			
City_Name:			
State_ID (1-36):			
City ID	City Name	City Name	State ID
1	Vizagapatnam	1	
2	Vijaywada	1	
3	Hansep	2	
4	Guwahati	3	
5	Patna	4	
6	Rajpur	5	
7	Panaji	6	
8	Ahmedabad	7	
9	Gurugram	8	
10	Shenzo	9	
11	Ranchi	10	
12	Bengaluru	11	
13	Theniyanthapuram	12	
14	Bhopal	13	
15	Mumbai	14	
16	Imphal	15	
17	Shillong	16	
18	Aizawl	17	
19	Kohima	18	
20	Bhubaneswar	19	
21	Chandigarh	20	
22	Jaipur	21	
23	Gangtok	22	
24	Chennai	23	
25	Hyderabad	24	
26	Agartala	25	
27	Lucknow	26	
28	Dehradoon	27	
29	Kolkata	28	
30	Pott Barr	29	
32	Dit	31	
33	Kavaratti	32	
34	Delhi	33	
35	Puducherry	34	
36	Srinagar	35	
37	Leh	36	
38	Jodhpur	21	
39	Ahmednagar	14	
40	Nashik	14	
41	Nagpur	14	

Employer, Employment, and City Management

Employer Employment City Health Facility

Facility_ID:	Facility_Name:	Facility_Type:	Bed_Capacity:
1	All India Institute of Medical Sciences	Hospital	800
2	Post Graduate Institute of Medical Education and Research	Hospital	1500
3	Tata Memorial Hospital	Cancer Hospital	600
4	Forts Hospital	Multi-Specialty Hospital	200
5	Apollo Hospitals	Multi-Specialty Hospital	400
6	Max Super Speciality Hospital	Multi-Specialty Hospital	300
7	Narayana Health	Multi-Specialty Hospital	1000
8	Medanta – The Medicity	Multi-Specialty Hospital	800
9	Kokilaben Dhirubhai Ambani Hospital	Multi-Specialty Hospital	300
10	Manipal Hospital	Multi-Specialty Hospital	500
11	Lilavati Hospital	Multi-Specialty Hospital	350
12	Sri Ramachandra Medical Centre	Hospital	700
13	Sankara Nethralaya	Eye Hospital	250
14	P.D. Hinduja Hospital	Multi-Specialty Hospital	300
15	Care Hospital	Multi-Specialty Hospital	350
16	Jaypee Hospital	Multi-Specialty Hospital	200
17	Staram Bharya Institute of Science and Research	Hospital	100
18	HCG Cancer Centre	Cancer Hospital	150
19	Aster CMI Hospital	Multi-Specialty Hospital	250
20	Asian Institute of Medical Sciences	Multi-Specialty Hospital	400
21	Narayana Institute of Cardiac Sciences	Heart Hospital	200
22	Apollo Spectra Hospitals	Surgical Hospital	150
23	Sahyadri Hospital	Multi-Specialty Hospital	300
24	MGM Healthcare	Multi-Specialty Hospital	400
25	MediHope Super Speciality Hospital	Multi-Specialty Hospital	200
26	BLK Super Speciality Hospital	Multi-Specialty Hospital	300
27	Asian Institute of Medical Sciences	Hospital	400
28	Sankle Hospital	Multi-Specialty Hospital	150
29	Jaslok Hospital	Multi-Specialty Hospital	600
30	Indraprastha Apollo Hospital	Multi-Specialty Hospital	500
31	Hiranandani Hospital	Multi-Specialty Hospital	300
32	Vivekananda Hospital	Hospital	250
33	Reddy Hospitals	Multi-Specialty Hospital	150
34	Forts Escorts Heart Institute	Heart Hospital	250
35	Yashoda Hospital	Multi-Specialty Hospital	400
36	Zydus Hospitals	Multi-Specialty Hospital	200
37	Vivekananda Hospital	Multi-Specialty Hospital	300
38	St. John's Medical College Hospital	Hospital	500
39	Birla Hospital	Multi-Specialty Hospital	150

Add Health Facility Update Health Facility Delete Health Facility Load Health Facility

MIND? MIND? MIND? MIND?

Employer, Employment, and City Management

Employer Employment City Health Facility

Facility_ID:	Facility_Name:	Facility_Type:	Bed_Capacity:
1	All India Institute of Medical Scie	Hospital	800
2	Post Graduate Institute of Medi	Hospital	1500
3	Tata Memorial Hospital	Cancer Hospital	600
4	Forts Hospital	Multi-Specialty Hospital	200
5	Apollo Hospitals	Multi-Specialty Hospital	400
6	Max Super Speciality Hospital	Multi-Specialty Hospital	300
7	Narayana Health	Multi-Specialty Hospital	1000
8	Medanta – The Medicity	Multi-Specialty Hospital	800
9	Kokilaben Dhirubhai Ambani H	Multi-Specialty Hospital	300
10	Manipal Hospital	Multi-Specialty Hospital	500
11	Lilavati Hospital	Multi-Specialty Hospital	350
12	Sri Ramachandra Medical Cen	Hospital	700
13	Sankara Nethralaya	Eye Hospital	250
14	P.D. Hinduja Hospital	Multi-Specialty Hospital	300
15	Care Hospital	Multi-Specialty Hospital	350
16	Jaypee Hospital	Multi-Specialty Hospital	200
17	Staram Bharya Institute of Sci	Hospital	100
18	HCG Cancer Centre	Cancer Hospital	150
19	Aster CMI Hospital	Multi-Specialty Hospital	250

Add Health Facility Update Health Facility Delete Health Facility Load Health Facility

5.2 Perform CRUD operations using GUI

1) Add Employers:

Before adding Employer to the table of employers

The screenshot shows a PostgreSQL client interface with the following details:

- Top Bar:** Dashboard, Properties, Statistics, Dependencies, Dependents, Processes, Youngsters_db/postgres@PostgreSQL 16*
- Toolbar:** Includes icons for New, Open, Save, Print, Copy, Paste, Find, etc.
- Query Editor:** Shows the SQL query: `select * from employer;`
- Data Output:** Displays a table with 98 rows of employer data. The columns are employer_id, employer_name, location, and salary_benefits.
- Employer Management Panel:** A modal window titled "Employer Management" with tabs for "Employer" and "Employment". It has input fields for Employer_ID, Employer_Name, Location, and Salary_benefits, and a "Salary_benefits" dropdown. Below the panel is a table with the same 98 rows of data.
- Bottom Status Bar:** Total rows: 98 of 98, Query complete 00:00:00.076, Ln 1, Col 23, ENG IN, 7:32 PM, 11/8/2024.

Employer_ID	Employer_Name	Location	Salary_benefits
88	FreeCharge	Gurgaon	650000
89	PayPal	Hyderabad	800000
90	PayU	Gurgaon	700000
91	Razorpay	Bengaluru	720000
92	Cure.fit	Bengaluru	650000
93	PharmEasy	Mumbai	700000
94	1mg	Bengaluru	680000
95	Netmeds	Hyderabad	640000
96	Zomato	Gurgaon	600000
97	Mongoneese	Gandhinagar	300000
98	Lions	Ahmedabad	300000

After adding the Employer to Employer table

The screenshot shows the DBeaver interface with the following details:

- Query Bar:** Shows the connection "Youngsters_db/postgres@PostgreSQL 16*" and the query "select * from employer;".
- Data Output:** Displays the results of the query in a table format. The columns are employer_id, employer_name, location, and salary_benefits. The data includes entries like PayPal (Hyderabad, 800000), PayU (Gurgaon, 700000), and Zomato (Gurgaon, 600000).
- Employer Management Panel:** A right-hand panel titled "Employer Management" with tabs for "Employer" and "Employment". It shows a summary row for Employer_ID 99 (Lucario, Bangalore, 500000) and a detailed table below it.
- Detailed Data Table:** Shows the same data as the main output table but with a different set of rows (89 to 99). The columns are employer_id, employer_name, location, and salary_benefits.
- Bottom Status:** Shows "Total rows: 99 of 99" and "Query complete 00:00:00.095".
- System Bar:** Shows the date and time as "11/8/2024 7:33 PM" and system status icons.

Before adding Employment to the table of employment

The screenshot shows the pgAdmin 4 interface with the 'Employer Management' window open. The 'Employer' tab is selected, showing fields for Job_ID, Industry_Type, Salary, Start_Date, and End_Date. The 'Employment' tab is also visible. Below the tabs is a table of 100 rows of employment data. A SQL query window at the top contains the command: 'select * from employment;'. The status bar at the bottom right indicates 'Ln 1, Col 25'.

Job_ID	Industry_Type	Salary	Start_Date	End_Date
91	User Interface Des...	70000	2022-09-01	2024-09-01
92	Network Administrat...	60000	2021-10-01	null
93	Health Informatics	70000	2023-11-01	
94	Pharmaceutical Sal...	75000	2020-12-01	2023-12-01
95	Regulatory Affairs	80000	2022-01-01	
96	Nonprofit Fundraising	40000	2023-02-01	2025-02-01
97	Training and Develop...	60000	2021-03-01	null
98	Corporate Social Res...	65000	2022-04-01	null
99	International Relation...	70000	2023-05-01	null
100	Business Intelligence	80000	2022-06-01	null

After adding Employment to the Employment table

The screenshot shows the pgAdmin 4 interface with the 'Employer Management' window open. The 'Employer' tab is selected, showing fields for Job_ID, Industry_Type, Salary, Start_Date, and End_Date. The 'Employment' tab is also visible. Below the tabs is a table of 101 rows of employment data. A SQL query window at the top contains the command: 'select * from employment;'. The status bar at the bottom right indicates 'Ln 1, Col 26'.

Job_ID	Industry_Type	Salary	Start_Date	End_Date
101	Manufacture	34000	2022-09-02	2024-09-01
92	Network Adminstrat...	60000	2021-10-01	
93	Health Informatics	70000	2023-11-01	
94	Pharmaceutical Sal...	75000	2020-12-01	2023-12-01
95	Regulatory Affairs	80000	2022-01-01	
96	Nonprofit Fundraising	40000	2023-02-01	2025-02-01
97	Training and Develop...	60000	2021-03-01	null
98	Corporate Social R...	65000	2022-04-01	null
99	International Relation...	70000	2023-05-01	null
100	Business Intelligence	80000	2022-06-01	null
101	Manufacture	34000	2022-09-02	2024-09-01

Before adding the city to city table

The screenshot shows the DBeaver interface with the 'City' tab selected in the top navigation bar. The 'Data Output' tab is active. A SQL query `select * from city;` is run, displaying 142 rows of city data. The columns are city_id, city_name, and state_id. The state_id column has a dropdown menu open, showing options like 1, 2, 3, etc. The last row of data is highlighted.

city_id	city_name	state_id
127	E2	26
128	67	9
129	74	35
130	66	27
131	112	28
132	114	21
133	118	14
134	119	27
135	120	8
136	121	26
137	122	21
138	124	14
139	132	14
140	134	35
141	139	11
142	142	32

After adding the city in City Table

The screenshot shows the DBeaver interface with the 'City' tab selected. The 'Data Output' tab is active. A SQL query `select * from city;` is run, displaying 142 rows of city data. The columns are city_id, city_name, and state_id. The state_id column has a dropdown menu open, showing options like 1, 2, 3, etc. The last row of data is highlighted.

city_id	city_name	state_id
128	67	9
129	74	35
130	66	27
131	112	28
132	114	21
133	118	14
134	119	27
135	120	8
136	121	26
137	122	21
138	124	14
139	132	14
140	134	35
141	139	11
142	142	32

Before adding the health facility in health_facility table

Facility_ID	Facility_Name	Facility_Type	Bed_Capacity
1	All India Institute of Medical Sciences	Hospital	800
2	Post Graduate Institute of Medical Education and Research	Hospital	1500
3	Tata Memorial Hospital	Cancer Hospital	800
4	Fortis Hospital	Multi-Specialty Hospital	200
5	Apollo Hospitals	Multi-Specialty Hospital	400
6	Max Super Speciality Hospital	Multi-Specialty Hospital	300
7	Narayana Health	Multi-Specialty Hospital	1000
8	Medanta - The Medicity	Multi-Specialty Hospital	900
9	Kokilaben Dhirubhai Ambani Hospital	Multi-Specialty Hospital	300
10	Manipal Hospital	Multi-Specialty Hospital	500
11	Lifeline Hospital	Multi-Specialty Hospital	350
12	Sri Ramachandra Medical Centre	Hospital	700
13	Sankara Nethralaya	Eye Hospital	200
14	P.D. Hinduja Hospital	Multi-Specialty Hospital	300
15	Care Hospital	Multi-Specialty Hospital	350
16	Jaypee Hospital	Multi-Specialty Hospital	200
17	Staran Shanti Institute of Social Medicine	Hospital	100
18	HCG Cancer Centre	Cancer Hospital	150
19	Other Health Centres	Multi-Specialty Hospital	300
Total rows: 100 of 100	Query complete 00:00:00.248	Ln 1, Col 20	

After adding the health facility to the health_facility table

Facility_ID	Facility_Name	Facility_Type	Bed_Capacity
101	Live Hospital	Women's Hospital	150
1	All India Institute of Medical Sciences	Hospital	800
2	Post Graduate Institute of Medical Education and Research	Hospital	1500
3	Tata Memorial Hospital	Cancer Hospital	800
4	Fortis Hospital	Multi-Specialty Hospital	200
5	Apollo Hospitals	Multi-Specialty Hospital	400
6	Max Super Speciality Hospital	Multi-Specialty Hospital	300
7	Narayana Health	Multi-Specialty Hospital	1000
8	Medanta - The Medicity	Multi-Specialty Hospital	900
9	Kokilaben Dhirubhai Ambani Hospital	Multi-Specialty Hospital	300
10	Manipal Hospital	Multi-Specialty Hospital	500
11	Lifeline Hospital	Multi-Specialty Hospital	350
12	Sri Ramachandra Medical Centre	Hospital	700
13	Sankara Nethralaya	Eye Hospital	200
14	P.D. Hinduja Hospital	Multi-Specialty Hospital	300
15	Care Hospital	Multi-Specialty Hospital	350
16	Jaypee Hospital	Multi-Specialty Hospital	200
17	Staran Shanti Institute of Social Medicine	Hospital	100
18	HCG Cancer Centre	Cancer Hospital	150
19	Other Health Centres	Multi-Specialty Hospital	300
Total rows: 101 of 101	Query complete 00:00:00.086	Ln 1, Col 28	

2) Update Students:

Before updating Employer to the table of employers

The screenshot shows the DBeaver interface with the 'Employer' tab selected in the 'Employer Management' panel. The 'Employer_ID' field contains the value '99'. The 'Employer_Name' field is empty. The 'Location' field is empty. The 'Salary_benefits' field is empty. Below the panel is a table of job data. At the bottom right of the interface, there are buttons for 'Add', 'Update', 'Delete', and 'Load'. The status bar at the bottom right shows 'Ln 1, Col 26', '7:32 PM', '11/6/2024', and a battery icon.

Employer_ID	Employer_Name	Location	Salary_benefits
99	PayPal	Hyderabad	600000
90	JD	Delhi	700000
91	Razorpay	Bengaluru	720000
92	Cure fit	Bengaluru	650000
93	PharmEasy	Mumbai	700000
94	Img	Bengaluru	680000
95	Netmeds	Hyderabad	640000
96	Dominos	Gurgaon	600000
97	Mongrel	Gandhinagar	300000
98	Lotts	Ahmedabad	300000
99	Lattice	Bangalore	500000

After updating Employer to the table of employers

The screenshot shows the DBeaver interface with the 'Employer' tab selected in the 'Employer Management' panel. The 'Employer_ID' field now contains the value '99'. The 'Employer_Name' field is filled with 'Latios'. The 'Location' field is filled with 'Gandhinagar'. The 'Salary_benefits' field is filled with '300000'. Below the panel is a table of job data. At the bottom right of the interface, there are buttons for 'Add', 'Update', 'Delete', and 'Load'. The status bar at the bottom right shows 'Ln 1, Col 26', '7:31 PM', '11/6/2024', and a battery icon.

Employer_ID	Employer_Name	Location	Salary_benefits
99	Latios	Gandhinagar	300000
90	JD	Delhi	700000
91	Razorpay	Bengaluru	720000
92	Cure fit	Bengaluru	650000
93	PharmEasy	Mumbai	700000
94	Img	Bengaluru	680000
95	Netmeds	Hyderabad	640000
96	Dominos	Gurgaon	600000
97	Mongrel	Gandhinagar	300000
98	Lotts	Ahmedabad	300000
99	Latios	Gandhinagar	300000

Before updating Employment to the table of employment

The screenshot shows the DBeaver interface with the 'Employment' table selected. The table has columns: Job_ID, Industry_Type, Salary, Start_Date, and End_Date. The data shows various job roles and their corresponding details. A modal window titled 'Employer Management' is open, showing fields for Job_ID, Industry_Type, Salary, Start_Date, and End_Date, all currently set to null.

Job_ID	Industry_Type	Salary	Start_Date	End_Date
87	Telecommunications Engineer	85000.00	2020-05-01	null
88	Cybersecurity Analyst	95000.00	2022-06-01	2025-06-01
89	Data Governance	75000.00	2021-07-01	null
90	Artificial Intelligence Research	95000.00	2023-08-01	null
91	User Interface Design	70000.00	2022-09-01	2024-09-01
92	Network Administration	60000.00	2021-10-01	null
93	Health Informatics	70000.00	2023-11-01	null
94	Pharmaceutical Sales	75000.00	2020-12-01	2023-12-01
95	Regulatory Affairs	80000.00	2022-01-01	null
96	Nonprofit Fundraising	40000.00	2023-02-01	2025-02-01
97	Training and Development	60000.00	2021-03-01	null
98	Corporate Social Responsibility	65000.00	2022-04-01	null
99	International Relations	70000.00	2023-05-01	null
100	Business Intelligence	80000.00	2022-06-01	null
101	Manufacture	34000.00	2022-09-02	2024-09-01

After updating Employment to the table of employment

The screenshot shows the DBeaver interface with the 'Employment' table selected. The table data remains the same as before. However, the 'Employer Management' modal window now shows updated values: Job_ID is 101, Industry_Type is 'Gold', Salary is 42000, Start_Date is 2020-01-01, and End_Date is 2023-02-03.

Job_ID	Industry_Type	Salary	Start_Date	End_Date
87	Telecommunications Engineer	85000.00	2020-05-01	null
88	Cybersecurity Analyst	95000.00	2022-06-01	2025-06-01
89	Data Governance	75000.00	2021-07-01	null
90	Artificial Intelligence Research	95000.00	2023-08-01	null
91	User Interface Design	70000.00	2022-09-01	2024-09-01
92	Network Administration	60000.00	2021-10-01	null
93	Health Informatics	70000.00	2023-11-01	null
94	Pharmaceutical Sales	75000.00	2020-12-01	2023-12-01
95	Regulatory Affairs	80000.00	2022-01-01	null
96	Nonprofit Fundraising	40000.00	2023-02-01	2025-02-01
97	Training and Development	60000.00	2021-03-01	null
98	Corporate Social Responsibility	65000.00	2022-04-01	null
99	International Relations	70000.00	2023-05-01	null
100	Business Intelligence	80000.00	2022-06-01	null
101	Manufacture	34000.00	2022-09-02	2024-09-01

Before updating the city in city

The screenshot shows a database management interface for the 'Youngsters_db' database. The main window title is 'Employer, Employment, and City Management'. The 'City' tab is selected. The 'City_ID' column header has a dropdown menu open, showing the current value '142' and a list of other city IDs: 142, 143, 31, 42, 52, 67, 74, 85, 112, 114, 118, 120, 121, 122, 124, 132, 134, 135, 136, 137, 138, 139, 140, 141, 142. Below this is a table with columns 'City_ID', 'City_Name', and 'State_ID'. The table contains 142 rows of city data. At the bottom right of the table are four buttons: 'Add City', 'Update City', 'Delete City', and 'Load City'. The status bar at the bottom right shows 'Ln 1, Col 19'.

After updating the city in city

The screenshot shows the same database management interface after an update. The 'City' tab is still selected. The 'City_ID' column header dropdown now shows the updated value '142' and a list of other city IDs: 142, 143, 31, 42, 52, 67, 74, 85, 112, 114, 118, 120, 121, 122, 124, 132, 134, 135, 136, 137, 138, 139, 140, 141, 142. The table below shows the same 142 rows of city data. The status bar at the bottom right shows 'Ln 1, Col 19'.

Before updating the health_facility of the health_facility table

The screenshot shows the DBeaver interface with a PostgreSQL connection to Youngsters_db. A query window displays the following SQL command:

```
select * from health_facility;
```

The results show a table with 101 rows. The last row (Facility_ID 101) has the following values:

Facility_ID	Facility_Name	Facility_Type	Bed_Capacity
101	Livestone Hospital	Multi-Specialty Hospital	340

Below the table, there are four buttons: Add Health Facility, Update Health Facility, Delete Health Facility, and Load Health Facility.

After updating the health facility of the health_facility table

The screenshot shows the DBeaver interface with a PostgreSQL connection to Youngsters_db. A query window displays the same SQL command:

```
select * from health_facility;
```

The results show the same table with 101 rows. The last row (Facility_ID 101) now has a different value:

Facility_ID	Facility_Name	Facility_Type	Bed_Capacity
101	Livestone Hospital	Multi-Specialty Hospital	340

The other rows remain identical to the previous screenshot.

3) Delete Students:

Before deleting employer from employer table

The screenshot shows the DBeaver interface with the 'Employer Management' window open. The 'Employer' tab is selected. A table displays employer data with columns: Employer_ID, Employer_Name, Location, and Salary_benefits. The data includes entries like MTNL, Google Pay, PhonePe, etc. Below the table are buttons for Add, Update, Delete, and Load. To the left, a SQL query window shows the command: 'select * from employer;'. The results of this query are displayed in a table below the interface, listing 99 rows of employer information.

Employer_ID	Employer_Name	Location	Salary_benefits
85	MTNL	Mumbai	500000
86	Google Pay	Hyderabad	720000
87	PhonePe	Bengaluru	700000
88	FreeCharge	Gurgaon	650000
89	PayPal	Hyderabad	800000
90	PayU	Gurgaon	700000
91	Razorpay	Bengaluru	720000
92	Cure fit	Bengaluru	650000
93	PharmEasy	Mumbai	700000
94	Tmg	Bengaluru	680000
95	Netmeds	Hyderabad	640000
96	Zomato	Gurgaon	600000
97	Mangonee	Gandhinagar	300000
98	Lions	Ahmedabad	300000
99	Latos	Gandhinagar	300000

After deleting the employer from employer table

The screenshot shows the DBeaver interface with the 'Employer Management' window open. The 'Employer' tab is selected. A table displays employer data with columns: Employer_ID, Employer_Name, Location, and Salary_benefits. The data includes entries like MTNL, Google Pay, PhonePe, etc. Below the table are buttons for Add, Update, Delete, and Load. To the left, a SQL query window shows the command: 'select * from employer;'. The results of this query are displayed in a table below the interface, listing 99 rows of employer information, identical to the previous screenshot but missing the deleted row (Employer_ID 99).

Employer_ID	Employer_Name	Location	Salary_benefits
85	MTNL	Mumbai	500000
86	Google Pay	Hyderabad	720000
87	PhonePe	Bengaluru	700000
88	FreeCharge	Gurgaon	650000
89	PayPal	Hyderabad	800000
90	PayU	Gurgaon	700000
91	Razorpay	Bengaluru	720000
92	Cure fit	Bengaluru	650000
93	PharmEasy	Mumbai	700000
94	Tmg	Bengaluru	680000
95	Netmeds	Hyderabad	640000
96	Zomato	Gurgaon	600000
97	Mangonee	Gandhinagar	300000
98	Lions	Ahmedabad	300000

Before deleting Employment to the table of employment

The screenshot shows the DBeaver interface with the 'Employer Management' window open. The 'Employment' tab is selected. A table lists various job positions with their industry type, salary, start date, and end date. One entry for 'Manufacture' (Job_ID 101) has its end date set to '2024-09-01'. The main SQL query window contains the command: `select * from employment;`. The status bar at the bottom right shows the date as 11/8/2024.

Job_ID	Industry_Type	Salary	Start_Date	End_Date
87	Telcommunications Enginee...	85000.00	2020-09-01	null
88	Cybersecurity Analysis	95000.00	2022-06-01	2025-06-01
89	Data Governance	75000.00	2021-07-01	null
90	Artificial Intelligence Research	95000.00	2023-08-01	null
91	User Interface Design	70000.00	2022-09-01	2024-09-01
92	Network Administration	60000.00	2021-10-01	null
93	Health Informatics	70000.00	2023-11-01	null
94	Pharmaceutical Sales	75000.00	2020-12-01	2023-12-01
95	Regulatory Affairs	80000.00	2022-01-01	null
96	Nonprofit Fundraising	40000.00	2023-02-01	2025-02-01
97	Training and Development	60000.00	2021-03-01	null
98	Corporate Social Responsibil...	65000.00	2022-04-01	null
99	International Relations	70000.00	2023-05-01	null
100	Business Intelligence	80000.00	2022-06-01	null
101	Manufacture	34000.00	2022-09-02	2024-09-01

After deleting Employment to the table of employment

The screenshot shows the DBeaver interface with the 'Employer Management' window open. The 'Employer' tab is selected. The table now shows the same data as before, but the entry for 'Manufacture' (Job_ID 101) has been removed, leaving a blank row where it previously was. The main SQL query window contains the command: `select * from employment;`. The status bar at the bottom right shows the date as 11/8/2024.

Job_ID	Industry_Type	Salary	Start_Date	End_Date
87	User Interface Des...	70000	2022-09-01	2024-09-01
88	Network Administrat...	80000	2021-10-01	
89	Health Informatics	70000	2023-11-01	
90	Pharmaceutical Sal...	75000	2020-12-01	2023-12-01
91	Regulatory Affairs	80000	2022-01-01	
92	Nonprofit Fundraisi...	40000	2023-02-01	2025-02-01
93	Training and Develop...	60000	2021-03-01	
94	Corporate Social R...	65000	2022-04-01	
95	International Relatio...	70000	2023-05-01	
96	Business Intelligence	80000	2022-06-01	
100				

Before deleting the city from city table

The screenshot shows the pgAdmin interface with a query editor containing the SQL command `select * from city;`. Below the query is a table titled "City" with columns: City_ID, City_Name, and State_ID. The table contains 142 rows of data. A specific row is selected for deletion, which is highlighted in the screenshot.

City_ID	City_Name	State_ID
142	Lavapur	32
141	Virend	7
31	Araku Valley	1
42	Dharmangar	25
52	Mirzapur	26
67	Kulu	9
74	Baramulla	35
85	Pithoragarh	27
112	Mirzapur	26
114	Bhwadi	21
118	Rishikesh	27
120	Sompat	8
121	Jhansi	26
122	Tork	21
124	Palghar	14
132	Amravati	14
134	Udhampur	35
139	Hassan	11
142	Badlapur	32

After deleting the city from city table

The screenshot shows the pgAdmin interface with the same query and table structure as the previous one. However, the row for "Lavapur" (City_ID 142) has been deleted, resulting in 141 rows of data.

City_ID	City_Name	State_ID
140	Kakradia	9
141	Virend	7
31	Araku Valley	1
42	Dharmangar	25
52	Mirzapur	26
67	Kulu	9
74	Baramulla	35
85	Pithoragarh	27
112	Mirzapur	26
114	Bhwadi	21
118	Rishikesh	27
120	Sompat	8
121	Jhansi	26
122	Tork	21
124	Palghar	14
132	Amravati	14
134	Udhampur	35
139	Hassan	11

Before deleting the health_facility of the Health facility table

The screenshot shows the pgAdmin interface with the 'Health Facility' table selected. The table contains 101 rows of data, each representing a hospital with its ID, name, type, and capacity. The data includes various hospitals like Liverstone Hospital, Srinivasa Hospital, and Jaypee Hospital.

Facility_ID	Facility_Name	Facility_Type	Bed_Capacity
101	Liverstone Hospital	Multi-Specialty Hospital	340
84	Jeevan Hospital	Multi-Specialty Hospital	300
85	Om Hospital	Multi-Specialty Hospital	500
86	Glenaeagles Global Health City	Multi-Specialty Hospital	700
87	Pyramid Hospital	Multi-Specialty Hospital	200
88	Sai Hospital	Multi-Specialty Hospital	100
89	Shubham Hospital	Multi-Specialty Hospital	200
90	Srinivasa Hospital	Multi-Specialty Hospital	300
91	MediHelp Hospital	Multi-Specialty Hospital	500
92	Eden Hospital	Multi-Specialty Hospital	600
93	Medicare Hospital	Multi-Specialty Hospital	250
94	Columbia Asia Hospital	Multi-Specialty Hospital	400
95	Pushpanjali Hospital	Multi-Specialty Hospital	250
96	Chaitanya Hospital	Multi-Specialty Hospital	100
97	Argyam Hospital	Multi-Specialty Hospital	300
98	Motherhood Hospital	Women's Hospital	200
99	Sunshine Hospital	Multi-Specialty Hospital	500
100	Jaypee Hospital	Multi-Specialty Hospital	400
101	Liverstone Hospital	Multi-Specialty Hospital	340

After deleting the health_facility of the Health_Facility table

The screenshot shows the pgAdmin interface with the 'Health Facility' table selected. The table now contains 100 rows of data, indicating that one row (Facility_ID 101) has been deleted. The remaining data is identical to the previous screenshot.

Facility_ID	Facility_Name	Facility_Type	Bed_Capacity
101	Liverstone Hospital	Multi-Specialty Hospital	340
84	Jeevan Hospital	Multi-Specialty Hospital	300
85	Om Hospital	Multi-Specialty Hospital	500
86	Glenaeagles Global Health City	Multi-Specialty Hospital	700
87	Pyramid Hospital	Multi-Specialty Hospital	200
88	Sai Hospital	Multi-Specialty Hospital	100
89	Shubham Hospital	Multi-Specialty Hospital	200
90	Srinivasa Hospital	Multi-Specialty Hospital	300
91	MediHelp Hospital	Multi-Specialty Hospital	500
92	Eden Hospital	Multi-Specialty Hospital	600
93	Medicare Hospital	Multi-Specialty Hospital	250
94	Columbia Asia Hospital	Multi-Specialty Hospital	400
95	Pushpanjali Hospital	Multi-Specialty Hospital	250
96	Chaitanya Hospital	Multi-Specialty Hospital	100
97	Argyam Hospital	Multi-Specialty Hospital	300
98	Motherhood Hospital	Women's Hospital	200
99	Sunshine Hospital	Multi-Specialty Hospital	500
100	Jaypee Hospital	Multi-Specialty Hospital	400

4) Load Students:

Loading the employers of employer table

The screenshot shows the DBeaver interface with the 'Employer Management' window open. The window has tabs for 'Employer' and 'Employment'. The 'Employer' tab is selected, showing a table with columns: Employer_ID, Employer_Name, Location, and Salary_Benefits. The data includes entries for various companies like Tata Consultancy Services, Infosys, Wipro, HCL Technologies, etc., with their respective locations and salary benefits. The 'Employment' tab is also visible but appears to be empty. Below the table, there are buttons for Add, Update, Delete, and Load. To the left, a SQL query is run in the 'Query' tab:

```
1 select * from employer;
```

The results table shows 99 rows of employer data. At the bottom, it says 'Total rows: 99 of 99 Query complete 00:00:00.065'.

Loading the employment of employment table

The screenshot shows the DBeaver interface with the 'Employment Management' window open. The window has tabs for 'Employer' and 'Employment'. The 'Employment' tab is selected, showing a table with columns: Job_ID, Industry_Type, Salary, Start_Date, and End_Date. The data includes various job roles and their details. The 'Employer' tab is also visible but appears to be empty. Below the table, there are buttons for Add Employment, Update Employment, Delete Employment, and Load Employment. To the left, a SQL query is run in the 'Query' tab:

```
1 select * from employment;
```

The results table shows 100 rows of employment data. At the bottom, it says 'Total rows: 100 of 100 Query complete 00:00:00.068'.

Loading the cities of city table

The screenshot shows the pgAdmin interface for a PostgreSQL database named 'Youngsters_db'. A query window on the left displays the SQL command: 'select * from city;'. The main pane shows a table titled 'City' with columns: City_ID, City_Name, and State_ID. The data grid lists 142 rows of city names and their corresponding IDs and state IDs. The bottom status bar indicates 'Total rows: 142 of 142' and 'Query complete 00:00:00.263'.

City_ID	City_Name	State_ID
1	Visakhapatnam	1
2	Vijayawada	1
3	Karur	2
4	Guwahati	3
5	Ponta	4
6	Rajput	5
7	Salem	6
8	Amaravati	7
9	Cuttack	8
10	Shimla	9
11	Ranchi	10
12	Bengaluru	11
13	Chennai	12
14	Indore	13
15	Mumbai	14
16	Imphal	15
17	Shillong	16
18	Azaad	17
19	Katima	18
20	Ehwalanagar	19

Loading the health facilities of the health_Facility table

The screenshot shows the pgAdmin interface for a PostgreSQL database named 'Youngsters_db'. A query window on the left displays the SQL command: 'select * from health_facility;'. The main pane shows a table titled 'Health Facility' with columns: Facility_ID, Facility_Name, Facility_Type, and Bed_Capacity. The data grid lists 100 rows of facility names and their types and capacities. The bottom status bar indicates 'Total rows: 100 of 100' and 'Query complete 00:00:00.248'.

Facility_ID	Facility_Name	Facility_Type	Bed_Capacity
1	All India Institute of Medical Sciences	Hospital	800
2	Post Graduate Institute of Medical Education and Research	Hospital	1500
3	Tata Memorial Hospital	Multi-Specialty Hospital	300
4	Fortis Hospital	Multi-Specialty Hospital	200
5	Apollo Hospitals	Hospital	400
6	Max Super Speciality Hospital	Multi-Specialty Hospital	300
7	Narayana Health	Hospital	1000
8	Medanta - The Medicity	Hospital	800
9	Manipal Hospitals Ambuja	Multi-Specialty Hospital	200
10	Manipal Hospital	Hospital	500
11	Lokay Hospital	Multi-Specialty Hospital	350
12	Sri Ramachandra Medical College Hospital	Hospital	700
13	Sankara Nethralaya	Eye Hospital	250
14	P.D. Hinduja Hospital	Multi-Specialty Hospital	300
15	Care Hospital	Hospital	350
16	Artemis Hospital	Multi-Specialty Hospital	200
17	Siddaram Bhadrappa Institute of Science and Technology	Hospital	100
18	HODS Cancer Centre	Cancer Hospital	150
19	Artemis CHI Hospital	Multi-Specialty Hospital	300

With the help of above diagrams we can visualize how are the CRUD Operations working in the tables of our database however we will document in brief of all the functions in the GUI:

- 1) The Add button is used in all the four tables of employer, employment, city and health_facility and for all of them we have the attributes of the corresponding table and using the values entered by the user we add (insert) the values in our table.
- 2) The update button is used in all the four tables of employer, employment, city and health_facility and for all of them we have the attributes of the corresponding table and using the values entered by the user we update the values in our table.
- 3) The delete button is used in all the four tables of employer, employment, city and health_facility and for all of them we have the attributes of the corresponding table and using the values entered by the user we delete the values in our table.
- 4) The load button is used in all the four tables of employer, employment, city and health_facility and for all of them we load all the values that are present in the table.

Chapter 6: Technical Issues and Solution

6.1 Technical Issues

1. Database Design Complexity

- Designing a normalized schema with multiple entities (e.g., Youngster, Migration Event, Employment) and relationships (e.g., one-to-many, many-to-many) required careful consideration to maintain data integrity and avoid redundancy.
- Normalizing the tables and avoiding the redundancies that occurred during the process was hectic and time consuming, but a new venture.

2. Foreign Key and Referential Integrity

- Implementing foreign keys correctly for relationships like FromRegion and ToRegion in the Migration_Event table was challenging due to the need to ensure data consistency without circular dependencies.

3. Data Population and Constraints

- Adding initial data to tables with strict constraints, especially NOT NULL constraints on foreign keys, caused insertion errors due to missing referenced data.

4. Handling Multivalued Attributes

- Storing multiple phone numbers for each youngster presented a challenge since this is a multivalued attribute.

5. Mapping States and Cities to other tables

- Mapping 100+ cities to their respective states and ensuring consistency across multiple tables like Climate was time-consuming and prone to errors.

6. Query and View Optimization

- Complex queries, especially joins and nested subqueries across multiple tables, resulted in slow performance.

7. GUI and Java Integration

- Building a GUI application in Java for CRUD operations required secure, efficient database handling.

6.2 Solution

1. Database Design Complexity

Approach Taken:

- **Step 1:** Began by identifying main entities (Youngster, Migration Event, Employment, Education, etc.) and defining their relationships, aiming for a BCNF-compliant design to reduce redundancy.
- **Step 2:** Used an Entity-Relationship Diagram (ERD) to visualize the relationships and dependencies among tables, which helped in spotting areas prone to anomalies.

-
- **Step 3:** Divided complex entities into smaller tables where needed and created linking tables (e.g., Enrolled for many-to-many relationships).

Tools Used: ERD design tool, SQL constraints for enforcing normalization rules.

Alternative Solution: Consider using a denormalized schema to reduce the number of tables and simplify queries. However, the normalized approach was chosen because it maintained data integrity and allowed more flexibility in future extensions.

2. Foreign Key and Referential Integrity

Approach Taken:

- **Step 1:** Identified relationships that required foreign keys, such as FromRegion and ToRegion in Migration_Event.
- **Step 2:** Carefully ordered table creation to avoid circular dependencies and used SQL constraints (FOREIGN KEY, ON DELETE CASCADE) to maintain referential integrity.
- **Step 3:** Performed test insertions and deletions to verify that referential integrity was preserved and that cascading deletions/updates worked as expected.

Tools Used:

PostgreSQL constraints and referential actions (ON DELETE CASCADE, ON UPDATE CASCADE).

Alternative Solution:

Considered a soft delete approach (flagging records instead of deleting) but found it less suitable since cascading deletions helped maintain data cleanliness when related records were removed.

3. Data Population and Constraints

Approach Taken:

- **Step 1:** To manage NOT NULL foreign key constraints, populated referenced tables (e.g., State, City) first to ensure necessary IDs existed.
- **Step 2:** Inserted data in a sequence that respected dependency hierarchy, inserting records into dependent tables (e.g., Youngster, Migration_Event) only after referenced records existed.
- **Step 3:** Used test cases to confirm that constraints were applied correctly and all required fields were populated without errors.

Tools Used: Batch insert scripts and SQL transactions to ensure atomicity during data population.

Alternative Solution: Considered temporarily removing constraints for faster insertion but chose not to, as this could have led to inconsistent data.

4. Handling Multivalued Attributes

Approach Taken:

- **Step 1:** Identified attributes that could have multiple values, such as Phone Numbers for each Youngster.
- **Step 2:** Created a separate table, `Youngster_Phone`, to store each phone number as a unique row associated with the Youngster's ID. This approach normalized the multivalued attribute.
- **Step 3:** Use a composite key (`Youngster_ID` and Phone Number) to enforce uniqueness.

Tools Used:

SQL schema design with composite primary keys.

Alternative Solution:

Considered storing phone numbers as a comma-separated list within a single column, but the normalized approach was chosen because it maintained flexibility for querying and indexing.

5. Mapping Regions and Cities

Approach Taken:

- **Step 1:** Collected accurate city-state mappings and inserted data for 100+ cities and 36 states using SQL.
- **Step 2:** Used scripts to automate data entry and ensure each city correctly mapped to a state, double-checking mappings using `SELECT` queries to validate accuracy.

-
- **Step 3:** Cross-referenced results by running sample queries to confirm consistency, particularly in Climate and Government_Policy tables.

Tools Used:

SQL scripts and verification queries.

Alternative Solution:

Manual data entry was considered but was dismissed due to the high chance of errors and time consumption.

6. Query Optimization for Performance

Approach Taken:

- **Step 1:** Identified frequently queried fields and applied indexing to primary and foreign keys, such as Youngster_ID, City_ID, and State_ID.
- **Step 2:** Rewrote subqueries as joins where possible, as joins often perform better in relational databases.

Tools Used:

Pgadmin4 for query analysis, indexes on key columns.

Alternative Solution:

Considered creating views for complex queries, but this was skipped due to time constraints and focus on direct optimization of the main queries.

7. GUI and Java Integration

Approach Taken:

- **Step 1:** Developed the dbmsProject class in Java to handle database connections securely with PreparedStatement for each SQL operation, protecting against SQL injection.
- **Step 2:** Modularized CRUD operations (e.g., insertEmployer, updateEmployer) for reusability and consistency across tables.
- **Step 3:** Tested the GUI thoroughly to confirm that each operation performed as expected and handled errors gracefully.

Tools Used:

Java Swing for GUI, PreparedStatement for secure SQL execution

Alternative Solution:

Considered using simple Statement objects but chose PreparedStatement for better security and efficiency.