Parte B: Información territorial basada en las comunas de Chile

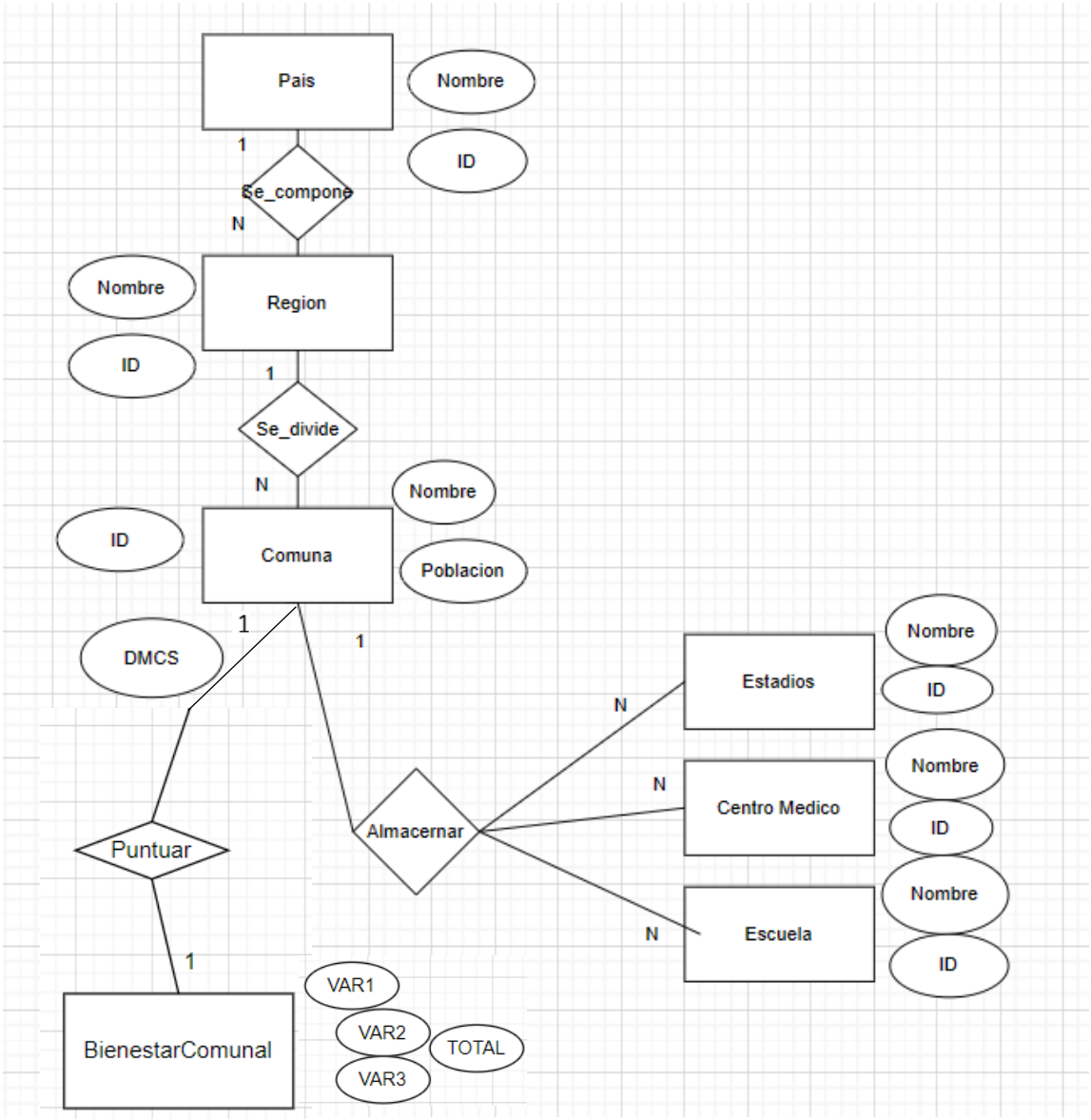
Integrantes: Cristóbal Felipe Rebolledo Oyanedel

Benjamín Enrique Parra Barbet

Carolina Andrea Obreque Higueras

Profesores: Luis Veas y Matthieu Vernier

Modelo Entidad-Relación:



Modelo Relacional:

Pais (PK_ID_Pais, Nombre)

Region (PK_ID_Region, Nombre_Region, FK_ID_Pais)

Comuna (PK_ID_Comuna, Nombre_Comuna , DMCS, Población , FK_ID_Region)

Estadios (PK_ID_Estadio, Nombre_Estadio, FK_ID_Comuna)

CentrosMedicos (PK_ID_Centro, Nombre_Centro, FK_ID_Comuna)

Escuelas (PK_ID_Escuela, Nombre_Escuela, FK_ID_Comuna)

BienestarComunal(VAR1,VAR2, VAR3, TOTAL, PK(FK_ID_Comuna))

Diccionario de Datos:

Nombre de tabla Pais

pk	Atributo	Tipo de dato	Tamaño	fk	tabla
x	ID_Pais	INT	6		
	Nombre_Pais	VARCHAR	30		

Nombre de tabla Region

pk	Atributo	Tipo de dato	Tamaño	fk	tabla
x	ID_Region	INT	6		
	Nombre_Region	VARCHAR	100		
	ID_Pais	INT	6	x	Pais

Nombre de tabla Comuna

pk	Atributo	Tipo de dato	Tamaño	fk	tabla
x	ID_Comuna	INT	6		
	Nombre_Comuna	VARCHAR	100		
	DMCS	FLOAT	32		
	Poblacion	INT	32		
	ID_Region	INT	6	x	Region

Nombre de tabla Estadios

pk	Atributo	Tipo de dato	Tamaño	fk	tabla
x	ID_Estadio	INT	6		
	Nombre_Estadio	VARCHAR	200		
	ID_Comuna	INT	6	x	Comuna

Nombre de tabla CentrosMedicos

pk	Atributo	Tipo de dato	Tamaño	fk	tabla
x	ID_Centro	INT	6		
	Nombre_Centro	VARCHAR	200		
	ID_Comuna	INT	6	x	Comuna

Nombre de tabla Escuelas

pk	Atributo	Tipo de dato	Tamaño	fk	tabla
x	ID_Escuela	INT	6		
	Nombre_Escuela	VARCHAR	200		
	ID_Comuna	INT	6	x	Comuna

Nombre de tabla BienestarComunal

pk	Atributo	Tipo de dato	Tamaño	fk	tabla
x	ID_Comuna	INT	6	x	Comuna
	VAR1	INT	6		
	VAR2	INT	6		
	VAR3	INT	6		
	TOTAL	INT	6		

Variables a utilizar

Las variables que vamos a utilizar en nuestra base de datos son entretención, salud, educación y seguridad.

De cada una de estas variables guardaremos su nombre, descripción del lugar, como ha cambiado al pasar los años y su valor. Las elegimos porque son importantes para cada comuna, ya que si un usuario desea informarse sobre una región específica querrá buscar información en un sitio simple y que abarque la información más importante, como son las variables que ocupamos, en donde sus datos específicos será la cantidad de locales, sus direcciones, disponibilidad de trabajo, la seguridad, educación para la familia, etc. Entonces al saber todos estos datos relevantes de cada comuna, podrá decidir fácilmente sobre qué comuna le conviene más vivir gracias a la ayuda de nuestra base de datos.

Documentar Proceso de Descarga

Para tablas Pais y Region se escribieron los datos manualmente.

Para conseguir todas las comunas de Chile, manualmente las copiamos en un archivo de texto, luego se hizo un programa de Python llamado procesaComunas.py para poder escribirlas en un formato de arreglo (según las instrucciones) para luego usar getData.py y transformarlo en un archivo csv.

Para métrica de entretención, nos dispusimos a calcular la cantidad de estadios por comuna, para obtener los datos accedimos a el siguiente link: Anexo:Estadios de fútbol de Chile - Wikipedia, la enciclopedia libre. Gracias a que los datos ya estaban correctamente tabulados, bastó hacer una simple importación con Excel para obtener la tablas. Ahora con los datos en la planilla, el mismo Excel nos permitió eliminar las columnas de más (Nos quedamos con ciudad, nombre y capacidad) y anexar todo, además de eliminar los paréntesis para poder tener el nombre de la comuna limpio en cada fila. Guardamos el archivo como un csv y usamos un pequeño programa de Python que llamamos quita.py el cual devuelve un formato correcto del texto. El código es el siguiente:

```
import unicodedata

import codecs

def remove_accents(text):

    return ''.join(c for c in unicodedata.normalize('NFD', text) if
unicodedata.category(c) != 'Mn').replace('"',"").replace("'","")

file = codecs.open("estadios_Salida.csv","r","utf-8-sig")

text = "".join(file.readlines())

output = codecs.open("estadios_Salida_SIN_TILDES.csv","w","utf-8-sig")

output.write(remove_accents(text))

file.close()

output.close()
```

Este código normaliza los caracteres UTF-8 y elimina las comillas. Guardamos el archivo de salida como estadios_Salida_SIN_TILDES.csv.

De esta manera, podemos usar el valor para comparar directamente cuantos estadios tienen dos comunas, o podríamos comparar manualmente estadios/habitantes para tener una métrica porcentual para comparar dos comunas con un número de habitantes muy distinto. De lo anterior no guardamos el número de habitantes, ya que la cantidad varía mucho año a año, entonces dependerá de la persona que quiera hacer las comparaciones estimar la población comunal (Podríamos agregar el valor habitante a la tabla comunas en una entrega futura si lo consideramos necesario).

Como métrica de seguridad, quisimos contabilizar los delitos DMCS (Delitos de mayor connotación social) tales como homicidio, lesiones, violación, robo con violencia, etc. Para poder hacer una correcta comparación tomaremos la tasa de incidencia cada 100 000 habitantes, de esta manera podemos comparar varias comunas sin importar su cantidad de habitantes.

Extrajimos la información de la plataforma pazciudadana.cl. Desde la pestaña de Ranking comunal se nos proporciona un acceso directo a estas estadísticas a través de un documento Excel. Lo descargamos directamente e hicimos un proceso similar al anterior:

1. Eliminamos las columnas que no aportaban a la investigación. (Nos quedamos con Comuna y Tasa cada 100mil)
2. Guardamos el archivo usando una extensión csv
3. Ejecutamos quita.py para normalizar el texto, eliminando tildes y comillas
4. Ahora tenemos un archivo csv listo para ser leído.

Para comparar la educación, accedimos a Planes y Programas de estudio – Datos Abiertos (mineduc.cl) y descargamos los planes y programas de estudios de 2022, entro de este archivo comprimido había un csv que contenía información acerca de todos los planes de estudios del país, pero lo que nos interesaba era principalmente la cantidad de escuelas por comuna, entonces había que descartar el resto de la información. Debido al gran tamaño del archivo nos vimos con dificultades, ya que ningún editor de texto común, ni tampoco Excel podían abrirlo sin producir una pérdida de datos, ya que debían cargar más de 200MB. La solución fue hacer un algoritmo en python que lo procesara procedural mente, para que solo queden las columnas que nos interesan (Nombre escuela y comuna) que elimina todas las instituciones educativas que ya hayan aparecido en el archivo. El algoritmo es el siguiente:

```
import codecs

output = codecs.open("planesYProgramas_Salida.csv","w","utf-8-sig")

with
codecs.open("20230307_Planes_y_programas_de_estudios_2022_20220131_PUBL.csv","
r","utf-8-sig") as file:

    file.readline() # Elimina la primera fila (Nombres de columnas)

    line = file.readline()

    nombresColegio = set()

    while (line):

        columnas = line.split(";")

        nombre = columnas[3]

        if(nombre not in nombresColegio):

            columnas_importantes = ";".join([columnas[3],columnas[10]])

            output.writelines(columnas_importantes)

            output.write("\n")

            nombresColegio.add(nombre)

        line = file.readline()

output.close()

print("Terminado!")
```

Luego, había que quitar los tildes y las comillas (si existían), por lo que usamos quita.py, resultando en la salida esperada. Además, colocamos los nombres de las columnas a mano, en la primera fila.

Como nota, puede que no se justifique tanto que DMCS sea una tabla aparte, pero su existencia hace que la base de datos sea más escalable, de esta manera si queremos que se guarde información adicional podemos modificar esta tabla sin afectar la tabla comuna

Por último, como métrica de salud decidimos ver la cantidad de centros médicos por comunas.

Obtuvimos la información a través del MINSAL más concretamente: Listado De Establecimientos (minsal.cl). Esta página contiene una tabla que tiene los datos que buscamos, los que son nombre y comuna. Ya que la información ya está tabulada, usamos Excel para importarla directamente desde la página al igual como hicimos previamente. En Excel, había errores en los que algunos establecimientos (cantidad diminuta) que estaban digitados dos veces, tenían el mismo nombre exacto y en la misma comuna, metrics.py se encarga de ignorar los duplicados así que no hubo problema.

Parte 2

Para obtener información relevante y confiable, se buscaron más fuentes para reemplazar los datos previos de la base de datos que se construyó en la parte 1. Lo primero, para corroborar los nombres de las comunas sin depender de la página poco fiable Wikipedia, se reemplazaron con los datos proporcionados por los datos proporcionados por el github del Ministerio de ciencias, tecnologia, conocimientos y educacion con datos obtenidos durante la pandemia. De este sitio se puede acceder a los datos que se buscaban en la carpeta input/DistribucionDEIS/baseFiles/DEIS_template.csv, los cuales son muy convenientes, ya que incluyen la población, nombre, código de región y comuna. Al obtener estos datos todos juntos ya no era necesario usar el csv que contenía las regiones ni las comunas. Los datos se copiaron en un archivo llamado datosComuna.csv y se eliminaron manualmente los datos de 'comuna_desconocida' y las filas que incluían los totales. Se usó metrics.py para cargarlos directamente a la base de datos, puesto que los datos ya estaban en ascii así que no se producirían problemas al importarlos.

Consideramos la tabla 'indicadores de bienestar' innecesaria ya que solo servía para indicar cual es la descripción del indicador. Las tablas fueron recreadas para adaptarse a los cambios que iban a hacerse, los cuales fueron: se agregaron varios campos ID para optimizar las búsquedas al evitar las comparaciones por caracteres reemplazando los PK, por ejemplo, en la tabla comuna, donde ahora el PK no es el nombre sino el código de la comuna, Además le agregamos el FK_ID_COMUNA las tablas Estadios, CentrosMédicos y Escuelas para conectarlas a la tabla Comuna. Otra modificación fue establecer el DMCS como un atributo de las comunas en lugar usar una tabla aparte.

Se modificó metrics para que cargara los valores con una consulta sql luego de ya haber creado la tabla comunas. Además, en la tabla comuna se agregó un atributo que indica la cantidad de habitantes, este se puede usar para sacar datos importantes en proporción de la población comunal.

Para cada métrica de indicador de bienestar (estadios, DMCS, etc), antes de crear las tablas se hizo una consulta en metrics de modo que, al recibir el nombre de la comuna, coloque el id que está en la tabla comuna ya creada.

Pese a que se quiso mejorar la calidad de los datos para los estadios, no se encontraron fuentes que recopilaron una gran cantidad de ellos de manera fiable, por lo que se siguió optando por usar Wikipedia para esta tabla. En general, los estadios top están muy bien documentados excepto por los estadios con capacidad <= 2000 (tabla 'otros estadios'), en cuyo caso, a veces faltaba información más allá de la capacidad, club, nombre y comuna. Aunque había otras fuentes, nunca se nombraban muchos estadios, por lo que decidimos arriesgarnos y usar los datos proporcionados por Wikipedia. De esto dejamos nota de que hace falta alguna fuente fiable que permita comprobar esta información.

Ya que ahora trabajamos con ID_Comuna en vez de usar los nombres como PK, se modificó ProcesarPlanesYProgramas.py y se volvió a procesar la tabla del rar, de esta manera la tabla nueva tabla consiste en (NOMBRE_COLEGIO; ID_COMUNA) lo que la hace mucho más fácil de insertar.

Objetivo: Determinar cuáles comunas son las mejores para vivir.

Métodos y resultados: Se determinaron las siguientes variables para poder clasificar una comuna:

- Tasa de crímenes (Solo DMCS): La comuna es considerada buena si y solo si su DMCS es superior a la media del país.
- Salud (VAR1): Una comuna gana un punto en salud si su cantidad de centros médicos / población es superior a la media del país.
- Entretención (VAR2): Una comuna gana un punto en entretención si la cantidad de estadios / población supera a la media del país.
- Educación (VAR3): Una comuna gana un punto en educación si su cantidad de escuelas/población es superior a la media del país.

De esto, se propone como mejor comuna a toda comuna que sea buena y tenga los 3 puntos mencionados anteriormente.

Para tabular estos datos se creó la tabla BienestarComunal que se define de la siguiente manera: BienestarComunal(PK(FK_ID_COMUNA), VAR1, VAR2, VAR3, TOTAL) con VAR1, VAR2, VAR3, TOTAL teniendo como valor por defecto 0. Usando un programa en Python llamado calculaIndices.py se hicieron las consultas correspondientes para filtrar en 1 y se insertaron esas filas, luego en el mismo programa se consulta por 2,3 y 4 y se hace un update para modificar sus respectivas variables y agregar 1 al total, más adelante en el informe, en el punto 5 de las 14 consultas se puede ver como es el proceso para determinar las comunas que cumplen con uno de esos puntos.

Para exportar estos datos desde la base de datos, se creó el programa exportarConsultasBienestar, de esta manera quedó tablaBienestarComunal toda la información con las columnas de la tabla BienestarComunal, y en tablaBienestarComunal los valores de cada comuna que cumplía con todos los parámetros con su respectivo nombre.

Para poder apreciar estos resultados desde una base de datos se actualizó el dump en la carpeta baseDeDatos, pero una forma de hacer manualmente este proceso es la siguiente:

1) Se crea la base de datos con el archivo SQL
2) Modificar metrics.py para que se conecte a la base de datos creada con el usuario y contraseña correctos, luego se ejecuta. Metrics poblará toda la base de datos con los datos de entrada.
3) Se ejecuta calculaIndices.py para poblar la tabla BienestarComunal y que queden guardados los resultados.

De esto, luego de ejecutar exportarConsultasBienestar.py para exportar los datos en un archivo de texto, algunas de las comunas seleccionadas y ordenadas por población fueron las siguientes (los campos de output eran NOMBRE_COMUNA, TOTAL_PUNTOS, POBLACION):

Coyhaique, Cauquenes, La Ligua, Canete, Panguipulli, Nueva Imperial, Rio Bueno, Illapel, Monte Patria, Nacimiento, Pitrufquen, Loncoche, Purranque y Paillaco.

Más adelante en el informe, en el apartado de conclusiones hablaremos más a fondo de esto, pero primero debemos mostrar algunas de las consultas SQL sobre las tablas las cuales proporcionan información la cual fue necesaria para poder llegar a estos datos, además de algunas consultas extra con información relevante.

Se hizo un programa llamado graficosPoblacionVsMejorComuna.py para graficar y comparar los intervalos de población vs la cantidad de comunas que aparecen como las mejores.

14 consultas SQL

1) **Cantidad de comunas:**
   SELECT count(*) AS cantidad_Comunas FROM Comuna;

2) **Nombre comuna y su ID:**
   SELECT NOMBRE_COMUNA, PK_ID_COMUNA AS ID_Comuna FROM Comuna;

3) **Calcula la media de DMCS:**

   SELECT sum(DMCS)/(select count(*) FROM Comuna) AS media FROM Comuna;

4) **Filtrar para sacar a todas las comunas cuyos DMCS sea mayor a la media del país.**

   SELECT * FROM Comuna c WHERE c.DMCS < (SELECT sum(DMCS)/(SELECT count(*) FROM comuna) AS media FROM Comuna);

5) **Filtrar para sacar todas las comunas cuya relacion CentroMedico/poblacion sea menor a la media:**
   SELECT PK_ID_COMUNA, totalCentros/poblacion
   AS centrosVsPoblacion from Comuna c join (Select FK_ID_COMUNA, count(*) AS totalCentros FROM CentrosMedicos cm GROUP BY FK_ID_COMUNA) c2 on c.PK_ID_COMUNA = c2.FK_ID_COMUNA HAVING centrosVsPoblacion > (SELECT count(*)/(select sum(c3.poblacion) FROM comuna c3) FROM CentrosMedicos) ORDER BY centrosVsPoblacion;

6) **Filtrar para sacar todas las comunas cuya relacion estadios/poblacion sea menor a la media:**

   SELECT PK_ID_COMUNA, totalEstadios/poblacion AS

   estadiosVsPoblacion FROM Comuna c join (Select FK_ID_COMUNA, count(*)

   AS totalEstadios FROM Estadios cm GROUP BY FK_ID_COMUNA) c2

   on c.PK_ID_COMUNA = c2.FK_ID_COMUNA having estadiosVsPoblacion >

   (select count(*)/(select sum(c3.poblacion) FROM comuna c3)

   FROM Estadios) ORDER BY estadiosVsPoblacion;

7) **Filtrar para sacar todas las comunas cuya relacion escuelas/poblacion sea menor a la media:**

```
SELECT PK_ID_COMUNA, totalEscuelas/poblacion AS
escuelasVsPoblacion FROM Comuna c join (Select FK_ID_COMUNA, count(*)
AS totalEscuelas FROM Escuelas cm GROUP BY FK_ID_COMUNA) c2
on c.PK_ID_COMUNA = c2.FK_ID_COMUNA having escuelasVsPoblacion >
(SELECT count(*)/(select sum(c3.poblacion) FROM comuna c3)
FROM Escuelas) ORDER BY escuelasVsPoblacion;
```

8) **Insertar una fila vacia (solo ID) en la tabla BienestarComunal:**

```
INSERT INTO BienestarComunal (FK_ID_COMUNA)  VALUES  ({insertar un id de comuna})
```

9) **Editar el campo VAR1 de BienestarComunal de una fila especifica:**

```
UPDATE BienestarComunal SET VAR1 = VAR1 + 1, TOTAL = TOTAL + 1 WHERE
FK_ID_COMUNA = {id de alguna comuna};
```

Otros ejemplos:

10) **Las 100 comunas (por codigo) que tienen más estadios:**

```
SELECT count(*), FK_ID_COMUNA FROM Estadios GROUP BY FK_ID_COMUNA ORDER BY
count(*) DESC LIMIT 100;
```

11) **Las 100 comunas (por codigo) que tienen más centros médicos:**

```
SELECT count(*) , FK_ID_COMUNA FROM CentrosMedicos GROUP BY FK_ID_COMUNA ORDER
BY count(*) DESC LIMIT 100 ;
```

12) **Las 100 comunas (por codigo) que tienen más escuelas:**

```
SELECT count(*), FK_ID_COMUNA AS ID_COMUNA FROM Escuelas GROUP BY
FK_ID_COMUNA ORDER BY count(*) DESC LIMIT 100;
```

13) **Las 50 comunas (por codigo) que tienen menos centros médicos:**

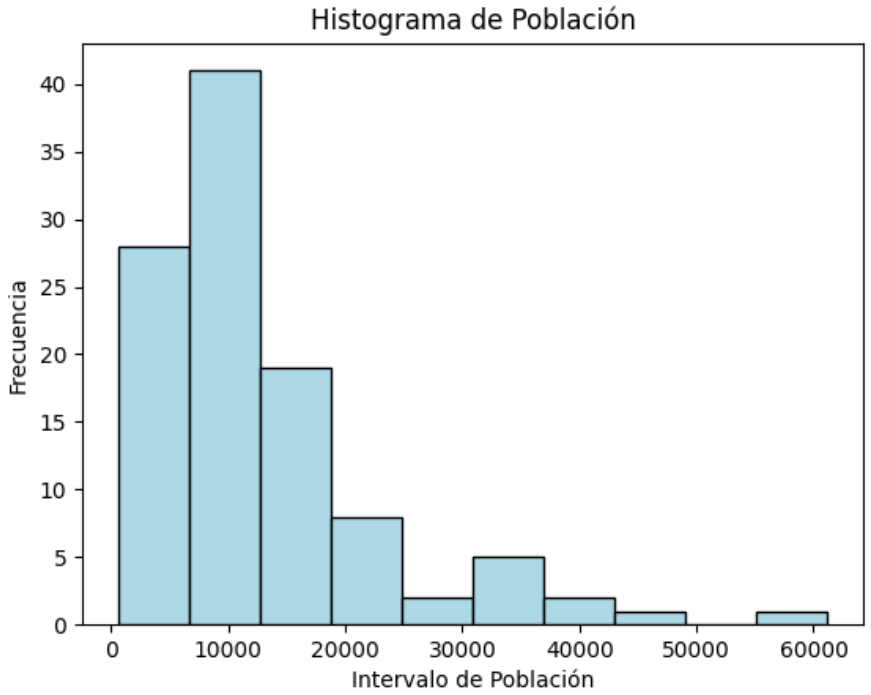
```
SELECT count(*) , FK_ID_COMUNA FROM CentrosMedicos GROUP BY FK_ID_COMUNA
ORDER BY count(*) ASC LIMIT 50;
```

14) **Las 50 comunas (por codigo) que tienen menos escuelas:**

```
SELECT count(*), FK_ID_COMUNA AS ID_COMUNA FROM Escuelas GROUP BY
FK_ID_COMUNA ORDER BY count(*) ASC LIMIT 50;
```

Conclusión y análisis:

Al analizar las tablas que creamos en base a los criterios anteriormente señalados, pudimos percatarnos que las comunas más grandes no siempre son las mejores para vivir, porque usualmente tienen una alta tasa de DMCS, primero creíamos que estas comunas eran las ideales para habitar ya que, si bien sabíamos que las comunas especialmente grandes podían tener una gran cantidad de crímenes DMCS, debido a su gran cantidad de población disminuiría. Si bien muchas comunas grandes cumplían con todos los estándares para ser las mejores calificadas en nuestros resultados, su alta tasa de criminalidad hizo que no fueran consideradas como comunas buenas, y debido a esto, se descartaron inmediatamente del top.



Al analizar este gráfico, se puede apreciar que, en comparación a su población, la probabilidad de que una comuna con menor cantidad de habitantes se encuentre entre las mejores para vivir es mayor, esto puede ser debido a dos motivos: Hay más comunas con poca gente, o que en general las comunas pequeñas tienen sus prioridades (educación, centros médicos, criminalidad y entretención) más cubiertas con relación a la población.

Las mejores comunas quedaron guardadas en el archivo comunasSeleccionadas.txt, el cual la primera columna es el nombre de la comuna, la segunda su puntuación y la tercera su población. Por otra parte, en el archivo tablaBienestarComunal.txt se muestran los datos guardados en la tabla BienestarComunal cuyas columnas indican ID_Comuna, Var1, Var2, Var3 y total.

Notas de la bitácora:

Al realizar este proyecto, en el proceso nos enfrentamos a demasiados imprevistos, los cuales tuvimos que solucionar a medida que íbamos avanzando en este proyecto, como, por ejemplo, muchas validaciones que no teníamos pensado que existían antes de comenzar nuestro trabajo. Creíamos que extraer datos de páginas webs era un tema más sencillo, pero gracias a este trabajo obtuvimos muchos conocimientos en bases de datos, consultas SQL, extracción e inserción de estos datos en páginas webs.

Con los diferentes resultados que obtuvimos, nuestro proyecto cumplió con el propósito principal, el cuál es que se simplificara la consulta de información a tal punto de que hasta un usuario cualquiera pueda revisar los resultados y ver cuáles fueron las mejores comunas respecto a nuestro parámetros.

Como nota para mejorar el análisis, y como propuesta, se pueden hacer mejores comparaciones con los datos de la tabla BienestarComunal y la información de la comuna, de manera que se puedan comparar mejor las estadísticas como, por ejemplo, población comunal vs puntos.

El último aspecto que llamó la atención fue la falta de información acerca de los estadios en Chile ya que no hay ninguna institución que se haya dedicado a recopilarlos, solo se obtuvo información de los estadios chicos a través de la página de Wikipedia, la cual como sabemos, puede ser manipulable. Nos gustaría que estos datos pudiesen ser verificados, pero eso ya se escapa de nuestras manos para esta entrega.