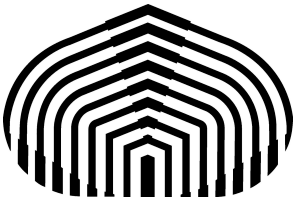


Universidad Simón Bolívar

Departamento de Computación y Tecnología de la Información

CI-3391 Laboratorio de

Sistemas de Bases de Datos I

Proyecto 1

Transporte de Personal

Estudiantes:

Maria Magallanes 13-10787

José Barrera 15-10123

Sartenejas, noviembre de 2019

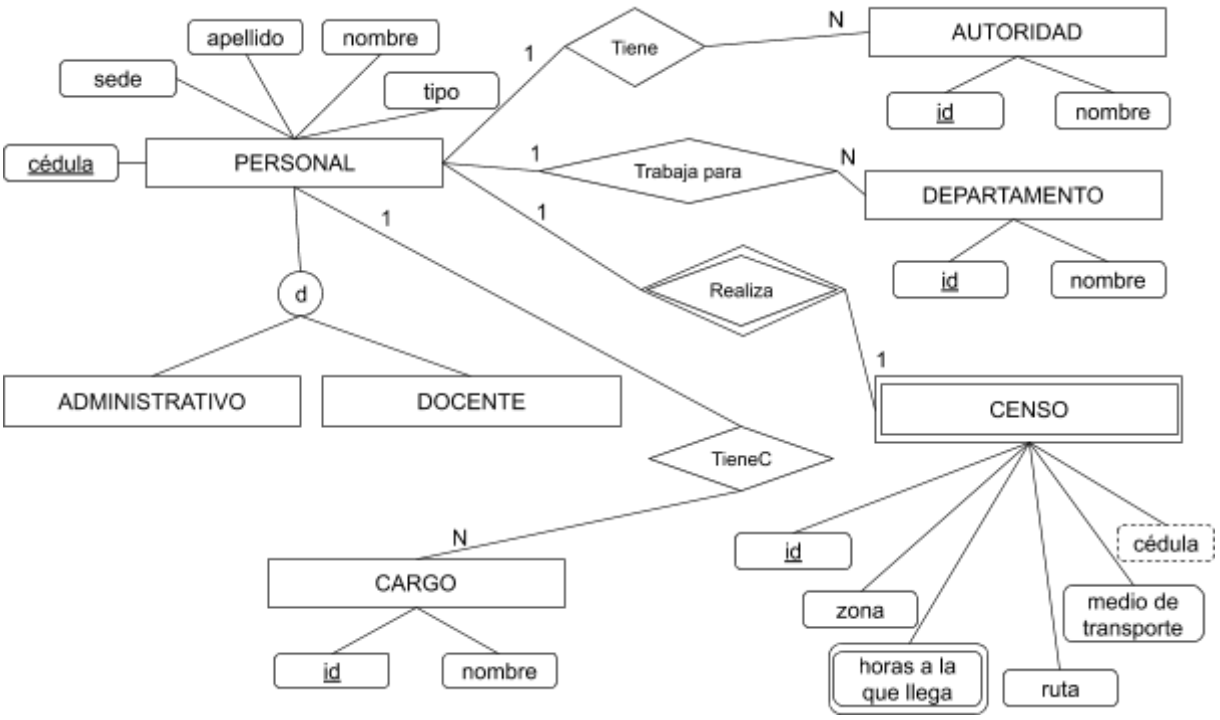
Introducción

La universidad Simón Bolívar ha tenido problemas para cubrir la demanda de transporte los últimos 3 meses; es por eso que se nos ha pedido que se modele y se diseñe una bases de datos que nos permita obtener información sobre el uso del mismo, para que así las autoridades correspondientes puedan tomar mejores decisiones; además es una forma de practicar los conceptos que hemos visto en clase de bases de datos.

En este proyecto se diseñará una base de datos que pueda almacenar de forma eficiente la información proporcionada por la nómina y el censo de la universidad; se implementará este diseño en PostgreSQL importando los datos proporcionados en un excel y, además presentaremos unas consultas SQL que muestran cómo obtener datos e información en la base de datos.

Creación de la Base de Datos

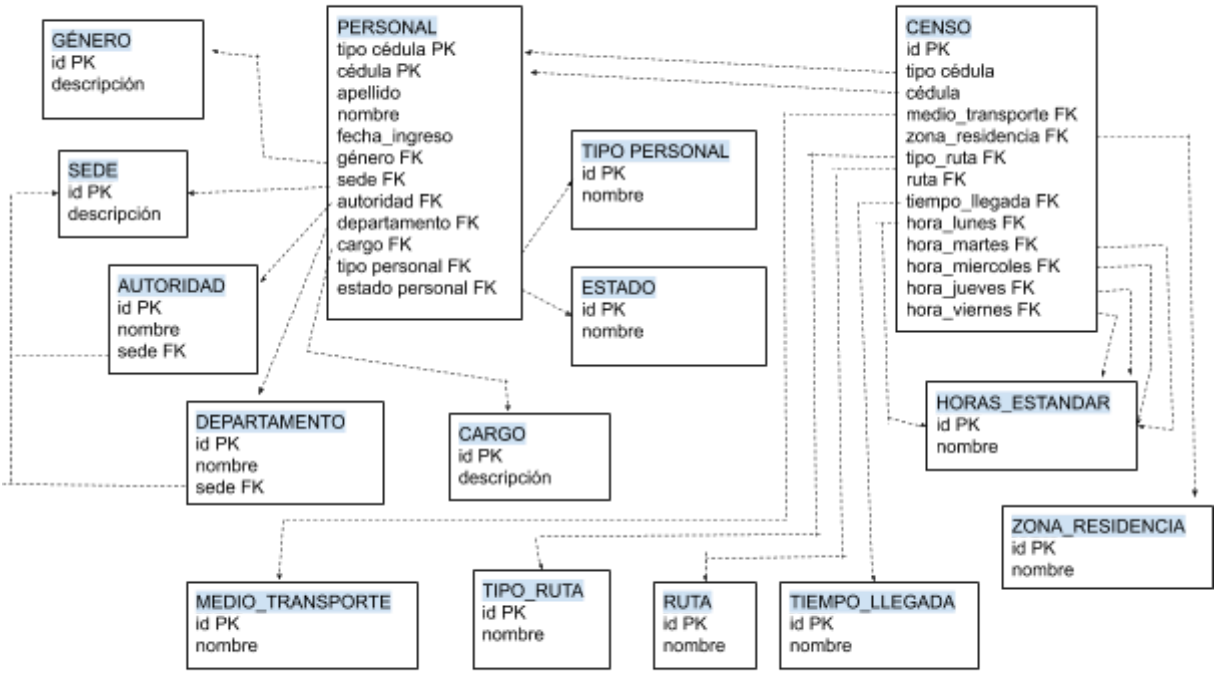
Para iniciar este proyecto se realizó un diagrama de Entidad Relación Extendido, con el fin de identificar las Entidades y las Relaciones que intervienen en el modelado de este problema.



Identificamos inicialmente 5 Entidades y 2 subentidades, que cada una tiene unos atributos particulares que obtuvimos de los datos proporcionados en el archivo de Excel y en el enunciado del proyecto.

Este modelo nos permitió tener más clara la estructura de nuestra base de datos.

Otra forma de verlo es con el diagrama de relación UML que se muestra a continuación:



En donde se muestran las relaciones y las entidades como contenedores en donde se identifica el PK (Primary Key) y se muestran las restricciones de sus atributos cuando estos son Foreing Key.

Para llegar a este diagrama se hicieron dos grandes entidades que son PERSONAL y CENSO, ya que tenía muchas relaciones 1:N que estas grandes entidades podían absorber. Sin embargo varios de los atributos por ser multivaluados, o estandarizados se les creó una nueva relación, que vienen a representar unas tablas de los distintos valores que pueden tomar ciertos atributos.

Para insertar los datos en la base

Para insertar la información administrada en la base de datos primero se pasaron los archivos de Excel a csv, lo que permitía obtener todo en formato plano que se pueda usar en la implementación en PostgreSQL.

En el archivo se crearon dos tablas temporales en donde se insertó toda la información que obtuvimos del csv con el comando COPY, para luego crear unas nuevas tablas que son las que se especifican en el modelo UML.

Se usaron también varias instrucciones de SQL que ayudaron con insertar directamente la información de una tabla a otra.

El comando INSERT propiamente, el comando INNER JOIN se usó bastante porque se tienen muchas claves foráneas, y el SELECT DISTINCT para evitar que los datos se repitieran.

También se tuvo cuidado de evitar información vacía en la tabla, por lo que se utilizó COALESCE para sustituir esas casillas por 'NR'

Consultas a la base de datos

Se asignaron 4 consultas:

1. ¿Quiénes son los empleados críticos que sabemos necesitan del transporte universitario en cada parada los jueves?

   Primero se determinó qué empleados son definidos críticos, y estos son los secretarios de los departamentos académicos (lo que tienen profesores), los asistentes

de estos departamentos, los profesores más antiguos y todos los técnicos de laboratorio.

Con esa información en mente se imprimió la ruta y la cédula del personal.

Para eso se utilizaron diversos INNER JOIN que permitan obtener toda la información en string (cadena de caracteres) lo que se necesitaba del personal que llenó el censo.

Con esta información se realizó un filtro con WHERE para que el medio de transporte a utilizar fuese el transporte universitario, y se filtró por usuarios que no contestaran NR o No usan el transporte USB, y por las rutas que ya están inactivas, además se tomaron personas del personal que cumplen con las restricciones de persona crítica, cosa que se resolvió con el operador OR; porque la persona crítica o es una secretaria de departamento académico, o una asistente o un técnico o el profesor con más antigüedad.

2. ¿Cuántos empleados críticos se pueden esperar adicionales?

Básicamente en esta consulta lo que se hace es volver a ver quienes son los empleados críticos para contarlos, utilizando la misma estrategia de restricciones que en la consulta anterior. Luego le restamos este valor a todos los empleados que son secretaria de departamento académico, o una asistente o un técnico o el profesor con más antigüedad, sin filtrar lo que respondieron en el censo.

3. ¿Cuántos empleados deben llegar los jueves en el transporte universitario a cada hora de cada departamento?

Para este query se imprime el nombre del departamento, la hora estándar y el número de empleados que deben llegar los jueves a esas horas específicas.
Para esto se realizaron diversos Join para juntar información relevante del personal con sus foreign keys, haciendo hincapié en que el medio de transporte de este personal

fuese efectivamente el transporte universitario y con el jueves siendo un un filtro para ver las horas de llegada de los empleados de esos departamentos, y claramente no pueden ser 'NR' (recordando que esto significa que dejaron vacío el censo), ni 'No usan transporte de la USB'.

Luego se agrupa y se cuenta por nombre de departamento y horas estándar, esto da la proporción de usuarios según el censo. Tras multiplicar por el número de personas en la nómina y multiplicarlo por 1/100, obtenemos la distribución de usuarios por parada de la nómina (asumiendo una distribución normal).

4. ¿Cuántos empleados totales se pueden esperar que usen el transporte universitario ese día?

Similar a la consulta anterior, se determinó cuántas personas utilizan el transporte USB, en un día dado, filtrando aquellos que respondieron que usan Transporte USB y que no respondieron NR o No uso transporte USB. A esta cantidad se le suma la probabilidad de que las personas que usan carro o con cola utilicen el transporte (que es ¼ por enunciado) . Luego por regla de tres; divido por el número de entradas del censo y se multiplica por el número de entradas de la nómina. Con esto se obtiene los empleados totales que se espera que utilicen el transporte de la USB

Resultados de las consultas 2-4 para todos los días

Query 2:

Lun	Mart	Mier	Juev	Vier
18	19	18	18	18

Query 4:

Lun	Mart	Mier	Juev	Vier
1892	1858	1913	1824	1831

En la consulta #2 referente a los empleados críticos adicionales, hay una mínima variabilidad en los datos. Siendo el máximo 19 y el valor que más se repite 18.

En la consulta #4 se observa que el dia donde podría esperarse una mayor demanda del transporte es el miércoles y la menor el jueves.

Para ejecutar la base de datos de este proyecto

La forma correcta de ejecutar la base de datos es entrando en la terminal y escribir:
> ./script_team5.sh

Con este comando ya tendremos la base con todos los datos necesarios para hacer consultas a ella. Es importante destacar que los archivos Nomina Empleados Sartenejas.csv y Censo Empleados.csv deben estar en la misma carpeta donde se encuentre el bd_team5.sql y el script.

Los queries del proyecto están en archivos distintos con la forma: query#N_team5.sql con N igual al número que identifica el query en cuestión.

Conclusiones

Las bases de datos fueron creadas y estandarizadas para simular hechos, y reglas de problemas del día a día, y con ellas se pueden modelar y consultar información que puede ser muy útiles y en principio quizá no tan evidentes. Por ejemplo en el caso particular de este proyecto, el saber diseñar y hacer consultas sobre la base de datos del personal nos puede permitir tomar mejores decisiones en cuanto a la planificación transporte.

En esta oportunidad se utilizó PostgreSQL que nos facilitó la inserción de los datos con su comando \COPY. Esta nos permitió de forma rápida utilizar los datos un csv para así llenar la base de datos.

El ON CONFLICT DO NOTHING también facilitó la implementación de la base, permitiendo llenar tablas problemáticas (por ejemplo: Horas estandarizadas)

COALESCE que nos permitió reemplazar espacios vacíos por NR, para evitar el campo NULL.

Debido a que se modulara fuertemente la base de datos para evitar la redundancia de datos y estandarizarlos, se tuvo que implementar muchas tablas y alguna de ellas con gran cantidad de Foreing Key. Esto causó que para hacer consultas se necesitarán varios JOIN.

Al final siempre tiene que tomar decisiones, y decidir cómo implementar una base de datos determina en gran parte de la complejidad, versatilidad, escalabilidad y eficiencia del sistema.