Universidad Nacional Autónoma de México

Facultad de ciencias

Proyecto:

Angel Christian Pimentel Noriega - 316157995
Mauricio Riva Palacio Orozco - 316666343
Alex Gerardo Fernandez Aguilar - 314338097
Martin Felipe Espinal Cruces - 316155362

Trabajo presentado como parte del curso de **Fundamentos de Bases de Datos** impartido por el profesor **Gerardo Avíles Rosas** .

1. **Modelo Entidad Relación:**
   Para la realización del modelo se tomaron en cuenta las siguientes consideraciones:

   - Se utilizó una superclase **Vehículo** con especialización para las entidades **Medio** que a su vez es la superclase de las especialidades **Microbús Trolebús RTP Autobús**, otra de la subclase de vehículo es la entidad **Masivo** la cual al igual que la entidad Medio es superclase pero de **Metrobús, Metro** y **Tren ligero**
   - Para los talleres se utilizó una entidad **Taller** la cual se relaciona con **Vehículo** a través de la relación **Reparar** la cual tiene los atributos fecha de ingreso, estimado de operación y razón.
   - Se definió la súper entidad **Usuario** para heredar todos sus elementos y relaciones a la sub entidad **Operador**, así podemos registrar el uso del transporte público de los operadores como lo hacemos con los usuarios.
   - Para la entidad **Usuario** se agregaron los atributos descritos en la especificación agregando la relación *Tiene* con la entidad **Licencia** la cual nos indica que tipo de vehículo puede manejar, junto con los datos específicos de cada licencia que el usuario tiene.
   - Se definió la pertenencia de Microbús, Trolebús, RTP y Autobús a la entidad **Ruta** a través de la superclase la cuál a su vez está relacionado con la entidad **Estación** por medio de la relación **Tiene** la cual tiene los atributos *fecha estimada* y *Disponible*.
     De manera similar se indicó la pertenencia de Metrobus, Metro y Tren ligero a la entidad **Linea** por medio de la relación **Pertenece** que se conecta con la superclase **Masivo**.
     Finalmente para la pertenencia de la entidad Taxi a la entidad **Sitio** por medio de la relación **Pertenece**
   - Para indicar que un operador debe realizarse un examen médico se creó la entidad **Examen Médico** que se relaciona con la entidad Operador por medio de la relación **Aplicar** que tiene los atributos *Hora* y *Fecha* para mantener registro del examen.
   - Para indicar la jornada del operador se agregaron los atributos necesarios a la relación **Manejar** estos atributos son: **Dia, Hora Fin y Hora Inicio**
   - Los métodos de pago heredan de una superentidad pago, la cual esta relacionada con la relación pagar, tal que los atributos son la cantidad y la fecha del pago. La herencia es absoluta, por lo que no se tendrá una tabla "Pagos"sino sera una tabla por cada método de pago.

2. **Modelo Relacional**:
   Al hacer la conversión del modelo entidad-relación al modelo relacional encontramos ciertas mejoras en el diseño y estos fueron los resultados:

   - El horario de los operadores, no depende únicamente del operador sino también del vehículo al manejarlo, entonces así el operador sabrá cuando le toca manejar el vehículo especificado. Este registro es único ya que un operador puede manejar el mismo vehículo mas de un día a la semana.
   - Con los usuarios, tendremos un registro cada vez que un usuario utiliza un vehículo. Ese registro es único. Debido a que esta tabla probablemente sea de las que tengan mas registros que cualquier otra, se podría fijar alguna regla la cual los registros mas viejos (mayor a 2 años) se vayan guardando paulatinamente en otra base de datos y se eliminen paulatinamente de la base de datos actual, así tendremos únicamente los registros de los viajes de los últimos 2 años, y debido a que el respaldo de los registros estará en otra base de datos (replica), no habrá un impacto en el desempeño de la base de datos.
   - En un futuro si se quisiera implementar un registro de cada vez que un operador manejó un vehículo, seria crear una tabla muy similar a la manejar, con el único cambio de día a fecha, así tendremos la fecha exacta con el horario en el que manejo el vehículo, además cambiar el nombre de la llave primaria, este registro será al igual que el de los usuarios un registro muy grande, pero menor al de los usuarios, por lo que propondría que en la misma replica en donde se guardan los datos de los viajes de hace mas de 2 años, se guarden paulatinamente los registros de los manejos de los vehículos mayores a 3 años, y paulatinamente se vayan eliminando de la base de datos.

- Las reparaciones de los vehículos también tienen una modificación, les agregamos una llave primaria única para poder diferenciar una reparación de otra debido a que sin esta no habría un registro de las reparaciones que se le hicieron a los vehículos en cierto taller. Ya que podría ser el caso que un mismo vehículo sea reparado en el mismo taller, y para evitar sobrescribir registros le agregamos el id de la reparación. Y cada reparación es única.
- Para definir la pertenencia de una ruta o una linea o ambas a una estación, se utilizaron tablas que usan llaves foráneas las cuales te dicen como están relacionadas.
- Agregar un nuevo método de pago es bastante sencillo, seria seguir la misma estructura de las tablas de débito/crédito y paypal, relacionar con el id del usuario y listo.
- El examen médico de un operador se tiene que hacer cada 6 meses, por lo que en la tabla del operador se agrego una columna de cada vez que se haga el examen médico se guarde la fecha de la ultima vez que se hizo el examen, así podremos saber si su ultimo examen médico esta vigente.
- El tipo de combustible de los vehículos, preferimos dejarlo como cadena, y no como una tabla de los tipos de combustible que podrían usar, debido a que no ganamos mucho al separarlo en una tabla, lo único que ocasionaríamos seria crear una tabla extra innecesaria, si fuera el caso que el combustible tuviera otras propiedades entonces si valdría la pena guardarlo en otra tabla, pero no es el caso.
- Gracias a que tenemos esta configuración, agregar un nuevo vehículo es muy sencillo, simplemente es crear una tabla como los demás vehículos, y definir si pertenece a un sitio, una ruta o una linea o si pertenece a algo distinto, crear las tablas pertinentes, sin embargo la mayor ventaja es que no se tiene que modificar nada en las tablas, solo es agregar tablas y relacionarlas.

3. **Dependencias Funcionales**:
   Las dependencias funcionales del Modelo Relacional son las siguientes:

   - Id_Usuario → Username, Contraseña, Email, Nombres, Apellido Paterno, Apellido Materno
   - Id_Operador → Id_Usuario, Fecha de Nacimiento, Género, Estudios, Dirección.
   - Id_Examen + Id_Operador → Cédula Médico, Peso, Presión, Talla, Status, Fecha, Hora.
   - Num_Pago + Id_Usuario → Número de Tarjeta, Código, Fecha de Vencimiento, Fecha del Pago, Cantidad del Pago, Usuario Paypal.
   - Id_Licencia + Id_Operador → Fecha de Vencimiento, Fecha de Expedición, Tipo.
   - Id_Vehiculo + Id_Linea → Fecha de Inicio, Capacidad de Pasajeros, Tipo de combustible.
   - Id_Vehiculo + Id_Ruta → Fecha de Inicio, Capacidad de Pasajeros, Tipo de combustible.
   - Id_Linea → Ubicación Inicio, Ubicación Fin, Nombre.
   - Id_Linea + Id_Estacion → Disponible, Fecha Estimada
   - Id_Ruta → Numero de Ruta, Distancia, Dirección Inicio, Direccion Final, Nombre.
   - Id_Linea + Id_Ruta → Disponible, Fecha Estimada.
   - Id_Estacion → Nombre, Hora Inicio, Hora Fin.
   - Id_Reparacion + Id_Vehiculo + Id_Taller → Fecha Estimada, Fecha de Ingreso, Razon.
   - Id_Taller → Direccion

4. **Normalización**:

   - Primer Forma Normal:
     Para llegar a la primera forma normal tuvimos que convertir los atributos multivaluados: **Teléfono** y **Especialidad** en sus propias tablas con llaves primarias los atributos: **Id Usuario** y **Id Taller** respectivamente.
     Las otras reglas de la Primer Forma Normal se cumplen desde la traducción del Modelo Entidad Relación

- Segunda Forma Normal:
  Por el diseño del modelo tenemos que no tenemos dependencias parciales ya que para todos los valores de las columnas de una fila dependen de la la llave primaria de dicha fila. Lo que es necesario y suficiente para que este modelo cumpla la segunda forma normal.
- Tercera forma normal:
  El diseño de este modelo relacional nos evita tener dependencias transitivas, esto es, no existe relaciones en el modelo tales que si A $\rightarrow$ B y B $\rightarrow$ C entonces A $\rightarrow$ C. Por lo tanto concluimos que este modelo relacional ya cumple la tercera forma normal y por tanto se encuentra libre (no en su totalidad ya que esto es imposible) de redundancias.

5. Integridad referencial, de dominio y de entidad.
   El modelo conserva la integridad referencial gracias a su manejo de llaves primarias y foráneas, así siempre que se encuentre una llave foránea en una tabla ésta llave aludirá a una fila válida de la tabla de la llave.
   Nuestro modelo cumple con integridad de dominio ya que en la creación de nuestra base de datos con base a queries de SQL especificamos que no se aceptan valores nulos en la llaves primarias ni foráneas junto con su tipo de valor.
   Por último conserva la integridad de entidad ya que el valor de cada llave primaria es único dentro de su propia tabla.

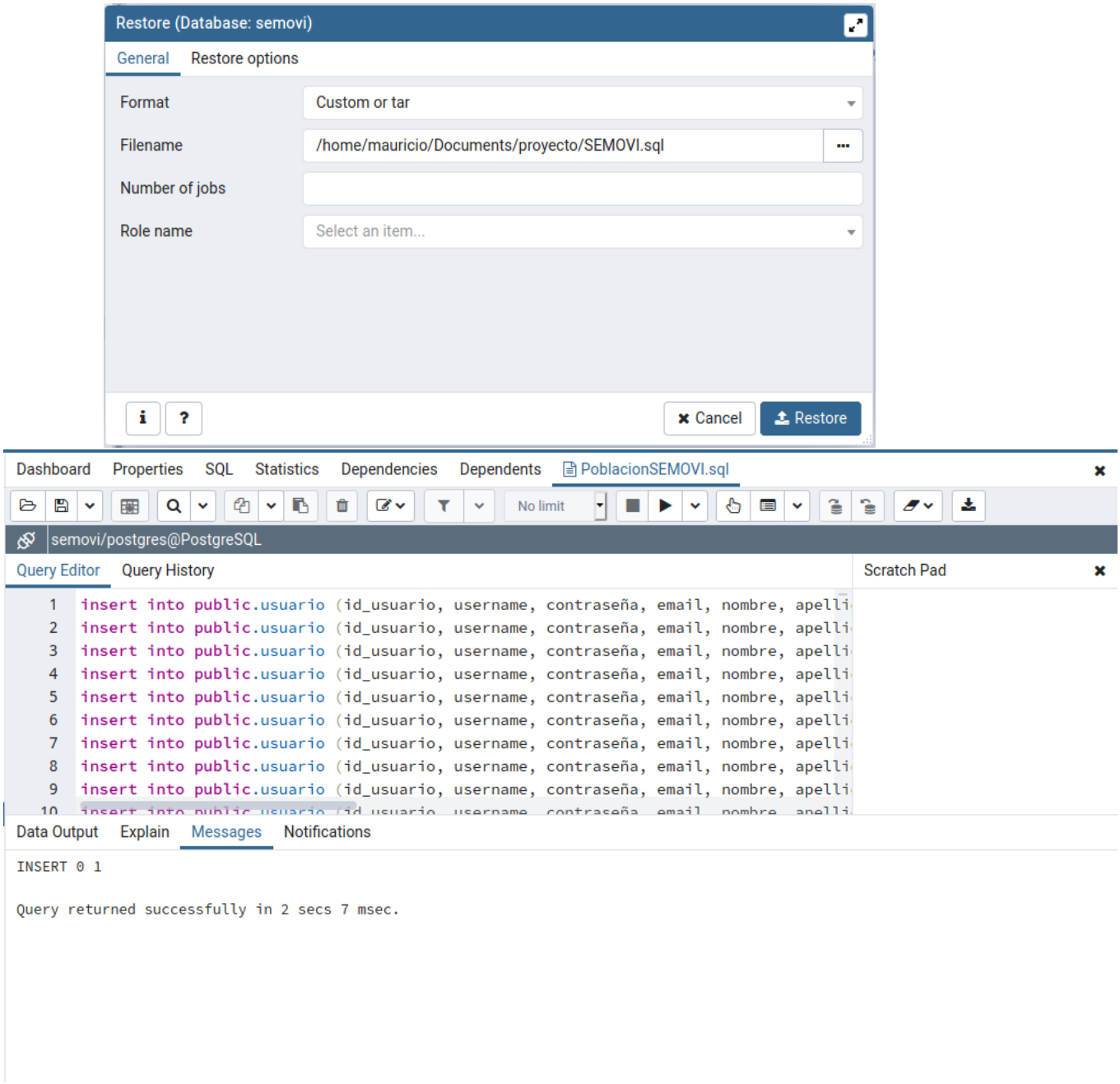
6. Errores de diseño:
   Nos encontramos con numerosos desafíos y uno de ellos fue la identificación de los distintos medios de transporte, en el modelo entidad relación propusimos una especialización disjunta para heredar los atributos que tienen en común los medios de transporte.
   Posteriormente al traducir al modelo relacional cometimos el error de generalizar la forma de identificar a los vehículos, es decir, todos los medios de transporte se identifican por un **id_vehiculo**, al finalizar la normalización no nos dimos cuenta de nuestro error.
   Después creamos la base de datos mediante el administrador pgAdmin, las tablas que causan conflicto son las siguientes:**Reparar, Utilizar y Manejar** el conflicto es el siguiente: las tablas anteriores tienen como llave foránea a **id_vehiculo** así que cada vez que queremos hacer una querie, por ejemplo, la reparación del metro **M-14** accede no sólo a la llave de el metro **M-14** si no que busca esa llave en cada medio de transporte lo cual resulta en el siguiente error:

   ERROR: insert or update on table reparar"violates foreign key constraint "fkis_vehiculo_autobus"
   DETAIL: Key (id_vehiculo)=(ME-70) is not present in table .ªutobus".

   Lamentablemente no contamos con el tiempo necesario para reiniciar el proceso de diseño con la esperanza que la documentación sea una retrospectiva de lo que no se debe hacer.

7. Instalación de la base de datos:

   - Para poder instalar de la base de datos, se tiene que crear una nueva base de datos, en este caso la nombre semovi, después de crear la base de datos dar click derecho y en Restore", seleccionar el archivo SEMOVI.sql y dar click en Restore".
   - Ahora damos click derecho otra vez sobre la base de datos y seleccionamos la opcion "Query Tool", dentro del recuadro derecho seleccionar el archivo "PoblacionSEMOVI.sqlz después dar click en el botón de "play".


Restore (Database: semovi)
General
Restore options
Format
Custom or tar
Filename
/home/mauricio/Documents/proyecto/SEMOVI.sql
Number of jobs
Role name
Select an item...
Cancel
Restore
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
PoblacionSEMOVI.sql
No limit
semovi/postgres@PostgreSQL
Query Editor
Query History
Scratch Pad
insert into public.usuario (id_usuario, username, contraseña, email, nombre, apelli
Data Output
Explain
Messages
Notifications
INSERT 0 1
Query returned successfully in 2 secs 7 msec.