

Institución Universitaria Pascual Bravo
Facultad de Ingeniería
Departamento de Sistemas Digitales

PROYECTO DE AULA (PA) - 2025-2 (TIA6)
DML - Lenguaje de Manipulación de Datos
EQUIPO "5"

Programa : Tecnología en desarrollo de Software
Asignatura : BASE DE DATOS I Código ET-0057
Docente : JAIME E SOTO U
Tipo proyecto : Aprendizaje Basado en Proyecto (ABP)

PROYECTO

Arquitectura de una base de datos para el servicio de Hospitalización del Sistema de Salud del Departamento de Antioquia, Colombia

Objetivo General del Proyecto

Los estudiantes de "Base de Datos I" deben diseñar, construir, poblar, consultar y validar una base de datos robusta, flexible y segura para almacenar y monitorear la información de hospitalización de la Red de Salud del Departamento de Antioquia, Colombia. La información de hospitalización debe estar integrada y alimentar a un Sistema de Historia Clínica Electrónica de cada paciente del Departamento de Antioquia. Para lograr esto, deben realizar las siguientes fases:

- **Tia 3: Modelo Lógico**
- **Tia 5: Implementación Base de Datos. Modelo Físico (DDL) (esta es la tarea a realizar)**
- **Tia 6: Manipulación de Base de Datos. DML - Lenguaje de Manipulación de Datos**
 - Fase 5: **Poblar la base de datos** (Insert)
 - Fase 6: **Construir el sistema de manipulación solicitado: Update, Delete, Select**
 - Fase 7: **Validar la Base de Datos con las propiedades ACID**

CONTEXTO

La Secretaría de Salud de la Gobernación de Antioquia tiene contemplada la licitación de un proyecto de envergadura para la atención en Salud en los servicios de Hospitalización de la Red De Atención del departamento. Inicialmente se requiere un sistema de información Web de Servicio de Atención e Información que se integrará en el futuro a un sistema de información de Historia Clínica Electrónica mucho más robusto. Para lograr este sistema de información, primero se debe construir una base de datos que cumpla con los requerimientos de la Secretaría de Salud para implementar un sistema de información de Hospitalización básico que se integre a un Sistema de Información de Historia Clínica Electrónica masivo.

Adicionalmente, desde el punto de vista técnico, la base de datos debe cumplir con las propiedades ACID. Las propiedades ACID garantizan que una transacción tenga fiabilidad, integridad y robustez en un sistema de gestión de bases de datos (DBMS). ACID es un acrónimo que representa Atomicidad, Consistencia, Aislamiento y Durabilidad.

- **Atomicidad (Atomicity)**: Asegura que todas las operaciones dentro de una transacción se realicen como una unidad. Si alguna parte de la transacción falla, la operación completa se revierte, y la base de datos vuelve a su estado anterior, como si la transacción nunca hubiera ocurrido.
- **Consistencia (Consistency)**: Garantiza que cualquier transacción lleve la base de datos de un estado válido a otro estado válido. Se asegura de que la integridad de los datos no se viole, manteniendo la coherencia estructural del sistema.
- **Aislamiento (Isolation)**: Asegura que las transacciones sean independientes entre sí. Esto significa que una transacción no debe afectar ni ser afectada por otras transacciones que se ejecutan simultáneamente, evitando la corrupción de datos y las lecturas inconsistentes.
- **Durabilidad (Durability)**: Garantiza que una vez que una transacción ha sido confirmada (commit), sus resultados son permanentes y persisten incluso en caso de fallas del sistema. Los cambios se guardan en el disco para asegurar que no se pierdan.

Institución Universitaria Pascual Bravo

Facultad de Ingeniería

Departamento de Sistemas Digitales

REQUERIMIENTOS

1.- REQUERIMIENTO GENERAL DE LA ACTIVIDAD

Poblamiento, Modificación, Eliminación y Consultas de una Bases de Datos que almacena la información del servicio de Hospitalización; como parte de la Historia Clínica Electrónica del Departamento de Antioquia, Colombia. En la actividad anterior, se realizó el proceso de “Definición de la Base de Datos Física”. En esta fase, se debe manipular la bases de datos del Modelo Físico a través de un Sistema de Gestión de Base de Datos (SGBD). El nombre de la base de datos es “[hce_antioquia](#)”

2.- REQUERIMIENTOS ESPECÍFICOS

- Poblamiento (INSERT) de todas las tablas del sistema tal como solicitado en el cuadro. Nota: puede utilizar el mecanismo de su preferencia para poblar las tablas. Debe seguir las indicaciones del cuadro suministrado.
- Actualización (UPDATE, DELETE). Después de poblar las tablas, debe realizar un conjunto de operaciones de actualización de la información de algunas tablas. Para esto, construirá instrucciones SQL - DML para actualizar y borrar registros.
- Construcción del sistema de consultas (SELECT)
 - Consultas simples y Complejas (JOIN)
 - Utilización de los elementos: GROUP BY, ORDER BY, HAVING, MAX, MIN, SUM, COUNT, AVG
- Construcción de vistas (VIEW)
- Validación de la Base de datos a través de los criterios ACID.
- Una vez culminada la tarea, se deben colocar todos los productos en el repositorio GIT que se viene utilizando durante el semestre. El GIT debe estar estructurado por Tareas y debe estar debidamente identificado como se ha solicitado desde el inicio del semestre.
- Anexo a esta plantilla de Informe, se le entrega una plantilla en Hoja de Cálculo para que coloque los resultados. En este informe cada estudiante miembro del grupo debe elaborar sus conclusiones individuales (mínimo de 300 palabras) sobre el impacto de esta tarea en su desarrollo académico y profesional; y cualquier reflexión que desee realizar sobre las competencias y saberes adquiridos.
- Video de Sustentación. En el video, se debe presentar con nombre e imagen cada miembro del grupo demostrando su participación en la tarea y evidenciando el código en ejecución.

3.- REQUERIMIENTOS DE DATOS

- Diccionario de Datos Físico
- Poblamiento de las tablas con Datos “simulados” de todas las tablas pero con criterios de coherencia

4.- REQUERIMIENTOS DE HERRAMIENTAS (debe utilizar estas herramientas)

- [Draw.io](#), Excel, PostgreSQL 15+, , pgAdmin4, Python (opcional)

5.- REQUERIMIENTOS DE ENTREGA DE PRODUCTOS (las entregas deben subirse al repositorio GIT)

- Poblamiento de las tablas del sistema de información con “data simulada” pero coherente (INSERT)
- Modificación de registros a través de actualización (UPDATE) y eliminación de registros (DELETE)
- Consultas básicas y avanzadas de información del sistema (SELECT)
- Validación de la base de datos a través de criterios ACID

Institución Universitaria Pascual Bravo

Facultad de Ingeniería

Departamento de Sistemas Digitales

INFORME DE ENTREGA

Tarea 5 (TIA5): Diccionario de Datos FÍSICO

BASES DE DATOS: “hce_antioquia”

Miembros del grupo

- Sebastian Sepulveda Quintero
- Andres Santiago Quiroz Gomez
- Emanuel Ceballos Mesa

1.- Poblamiento de la Base de datos (INSERT)

– El poblamiento se hizo mediante un archivo de python generado mediante prompt’s en claude, se editaba y se revisaba rigurosamente para que generará correctamente todos los datos y requerimientos específicos. No se utilizó los comandos nativos de postgresql porque para generar esta cantidad de registros sobrepasaba la capacidad de la ia ya que debían de ser datos fijos, y la generación aleatoria de los comandos de postgresql es muy limitada, así que se prefirió usar python y librerías de generación aleatoria.

Tabla	Cantidad de registros	Distribución	
Paciente	100	50 hombres, 50 mujeres. Divididos en 5 grupos etarios 10 registros (5 hombres, 5 mujeres) para cada grupo <ul style="list-style-type: none">• 1-10 años 10 registros• 11-20 años 15 registros• 21-40 años 20 registros• 41-60 años 25 registros• 61-+ años 30 registros Debe introducir un registro de paciente repetido. Error: equivocación en el número de documento de identificación	
Médico	30	16 hombres, 14 mujeres. Entre el total de 30 médicos deben haber 10 especialidades mínimo. Puede distribuirlos entre hombres y mujeres a placer. Nota: Por ejemplo, la especialidad en “Traumatología” puede tenerla tanto un hombre como una mujer. Debe introducir un registro de médico o médica repetido. Error: equivocación en el número de documento de identificación	
Especialidad Médica	10	No aplica el sexo o género	
Enfermera	10	8 mujeres, 2 hombres. Debe introducir un enfermero o enfermera repetido. Error: equivocación en el número de documento	
Hospital	10	No aplica el sexo o género	
Hospitalizacion	100	64 Pacientes con 1 hospitalización	64 registros
		10 Pacientes con 2 hospitalizaciones	20 registros (10 hombres, 10 mujeres)
		5 Pacientes con 3 hospitalizaciones	15 registros (8 hombres, 7 mujeres)
		1 paciente con 4 hospitalizaciones	1 registro (mujer)

Institución Universitaria Pascual Bravo
Facultad de Ingeniería
Departamento de Sistemas Digitales

2.- Actualización de registros de la Base de Datos (UPDATE)

	Tabla	Registros	Distribución
2.1	Pacientes	5	<i>* Actualizar la información de 5 registros de la tabla pacientes * Una actualización de UN SOLO campo en cada uno de los 5 registros * Debe seleccionar 5 campos diferentes. No debe repetir un mismo campo * Justifique la actualización. Nota: Escenario simulado/hipotético.</i>
2.2	Médicos	5	<i>Idem al ítem 2.1. pero con médicos</i>
2.3	Especialidades Médicas	5	<i>Idem al ítem 2.1. pero con especialidades</i>
2.4	Enfermeras	5	<i>Idem al ítem 2.1. pero con enfermeros/ras</i>
2.5	Hospitales	5	<i>Idem al ítem 2.1. pero con hospitales</i>
2.6	Hospitalizaciones	5	<i>Idem al ítem 2.1. pero con hospitalizaciones</i>

3.- Eliminación de registros de la Base de datos (DELETE)

Tabla	Cantidad de registros	Distribución
Pacientes	5	<i>Eliminar 5 pacientes</i>
Médicos	5	<i>Eliminar 5 médicos</i>
Especialidades Médicas	2	<i>Eliminar 2 especialidades médicas</i>
Enfermeras	5	<i>Actualizar la información de 10 campos del registro pacientes. No debe repetir el mismo campo en cada caso. Justifique su respuesta de porqué modificó el dato. Recuerde que es una situación hipotética.</i>
Hospitales	5	<i>Actualizar la información de 10 campos del registro pacientes. No debe repetir el mismo campo en cada caso. Justifique su respuesta de porqué modificó el dato. Recuerde que es una situación hipotética.</i>
Hospitalizaciones	5	<i>Actualizar la información de 10 campos del registro pacientes. No debe repetir el mismo campo en cada caso. Justifique su respuesta de porqué modificó el dato. Recuerde que es una situación hipotética.</i>

4.- Elaboración de consultas básicas (SELECT / Sin JOIN)

5.- Elaboración de consultas avanzadas (SELECT / Con JOIN)

6.- Creación y uso de Vistas (VIEW)

7.- Validación de la Base de Datos (ACID)

8.- Conclusiones individuales

- **Andres Quiroz**

Después de haber trabajado con todas estas operaciones de SQL, desde los UPDATE más básicos hasta las validaciones ACID más complejas, me doy cuenta de que esto va mucho más allá de solo escribir código. Al principio pensaba que las bases de datos eran simplemente guardar información en tablas, pero ahora entiendo que hay toda una arquitectura y lógica detrás que garantiza que los datos estén seguros, sean consistentes y estén disponibles cuando se necesiten.

Las consultas básicas sin JOIN me enseñaron a dominar los fundamentos: usar GROUP BY para agrupar datos, ORDER BY para organizarlos, y las funciones de agregación como COUNT, SUM, AVG, MAX y MIN para obtener estadísticas útiles. Pero cuando pasé a las consultas avanzadas con múltiples JOIN, fue cuando realmente vi el poder de las bases de datos relacionales. Conectar 5 o 6 tablas en una sola consulta para obtener un reporte completo de hospitalización con paciente, médico, especialidad, hospital y diagnóstico me hizo entender por qué el diseño del modelo entidad-relación fue tan importante desde el principio. Si no hubiera normalizado bien las tablas desde la fase de diseño, estas consultas serían imposibles o llenas de datos redundantes.

La creación de vistas fue un punto de inflexión para mí. Al principio no entendía por qué necesitábamos vistas si podíamos escribir las consultas directamente, pero después de crear la vista `vista_hospitalizacion_completa` con 6 JOIN y luego hacer 5 consultas diferentes sobre ella, vi la ventaja: escribes la lógica compleja una sola vez y luego la reutilizas. Es como tener un atajo que simplifica todo el trabajo futuro. Las vistas son especialmente útiles cuando trabajas en equipo porque otros desarrolladores pueden usar tu vista sin necesidad de entender toda la complejidad de los JOIN que hay detrás.

Si tuviera que resumir lo más importante que aprendí, diría que es el respeto por los datos. Antes veía las bases de datos como algo mecánico, pero ahora entiendo que son la memoria institucional de una

Institución Universitaria Pascual Bravo
Facultad de Ingeniería
Departamento de Sistemas Digitales

organización. Cada INSERT, UPDATE o DELETE tiene un significado y consecuencias. Las transacciones no son solo sintaxis, son la manera de garantizar que los cambios sean seguros. Las vistas no son solo conveniencia, son herramientas para compartir conocimiento estructurado. Y ACID no son solo cuatro propiedades abstractas, son el contrato de confianza entre la aplicación y la base de datos.

- **Sebastian Sepulveda Quintero**

Honestamente, cuando empecé este proyecto no tenía ni idea de que las bases de datos fueran tan complejas. Pensaba que era solo hacer tablas, meter datos y ya. Pero después de trabajar con todos estos scripts de UPDATE, DELETE, consultas básicas, avanzadas, vistas y validaciones ACID, me di cuenta de que hay todo un mundo detrás de cada operación que hacemos en una base de datos.

Las vistas fueron otro descubrimiento importante. Crear esa super vista con 6 JOIN fue complicado, no voy a mentir, pero después cuando hice las 5 consultas sobre ella fue súper fácil. Es como si hubiera creado un acceso directo a información compleja. Ya no tenía que escribir los 6 JOIN cada vez, solo usaba la vista y listo. Esto me hizo pensar en lo práctico que sería en un trabajo real: creas la vista una vez y todo el equipo puede usarla sin complicarse la vida.

Lo de ACID, las validaciones entendí por qué es tan importante. Atomicidad significa que o se hace todo o no se hace nada - no hay medias tintas. Cuando intenté actualizar ese paciente con ID 20 que no existía y me salió el error, al principio me frustré, pero luego entendí que ese error era exactamente lo que debía pasar. El sistema me estaba protegiendo de meter la pata. Eso es Consistencia en acción: la base de datos no me deja hacer cambios que rompan las reglas.

Académicamente este trabajo me ayudó un montón. Ya no veo SQL como solo memorizar comandos, sino como una herramienta poderosa para manejar información compleja. Las funciones de agregación como COUNT, SUM, AVG, MAX y MIN ya no son solo teoría del libro, ahora sé cuándo y cómo usarlas. GROUP BY y ORDER BY se volvieron mis mejores amigos para organizar datos. Y lo mejor es que ahora puedo leer consultas SQL complejas que antes me parecían jeroglíficos.

Profesionalmente esto me abre puertas. En cualquier empresa hay bases de datos, y saber manejarlas bien te hace valioso. No es solo saber hacer un SELECT simple, es entender transacciones, validaciones, integridad de datos. Cuando vaya a una entrevista y me pregunten sobre ACID o sobre cómo manejaría una operación crítica en una base de datos, ya no me quedo en blanco. Tengo ejemplos reales que hice, errores que cometí y aprendí a resolver.

- **Emanuel Ceballos Mesa**

Las consultas básicas sin JOIN parecían fáciles al principio, solo usar SELECT con ORDER BY y funciones como COUNT o AVG, pero cuando tuve que pensar qué información era realmente útil para el sistema de hospitalización, me di cuenta que no es solo escribir código sino entender qué necesita saber alguien que trabaja en un hospital. Por ejemplo, saber cuántos pacientes tiene cada EPS o cuál es la edad promedio puede ayudar a tomar decisiones reales sobre recursos y personal

Institución Universitaria Pascual Bravo
Facultad de Ingeniería
Departamento de Sistemas Digitales

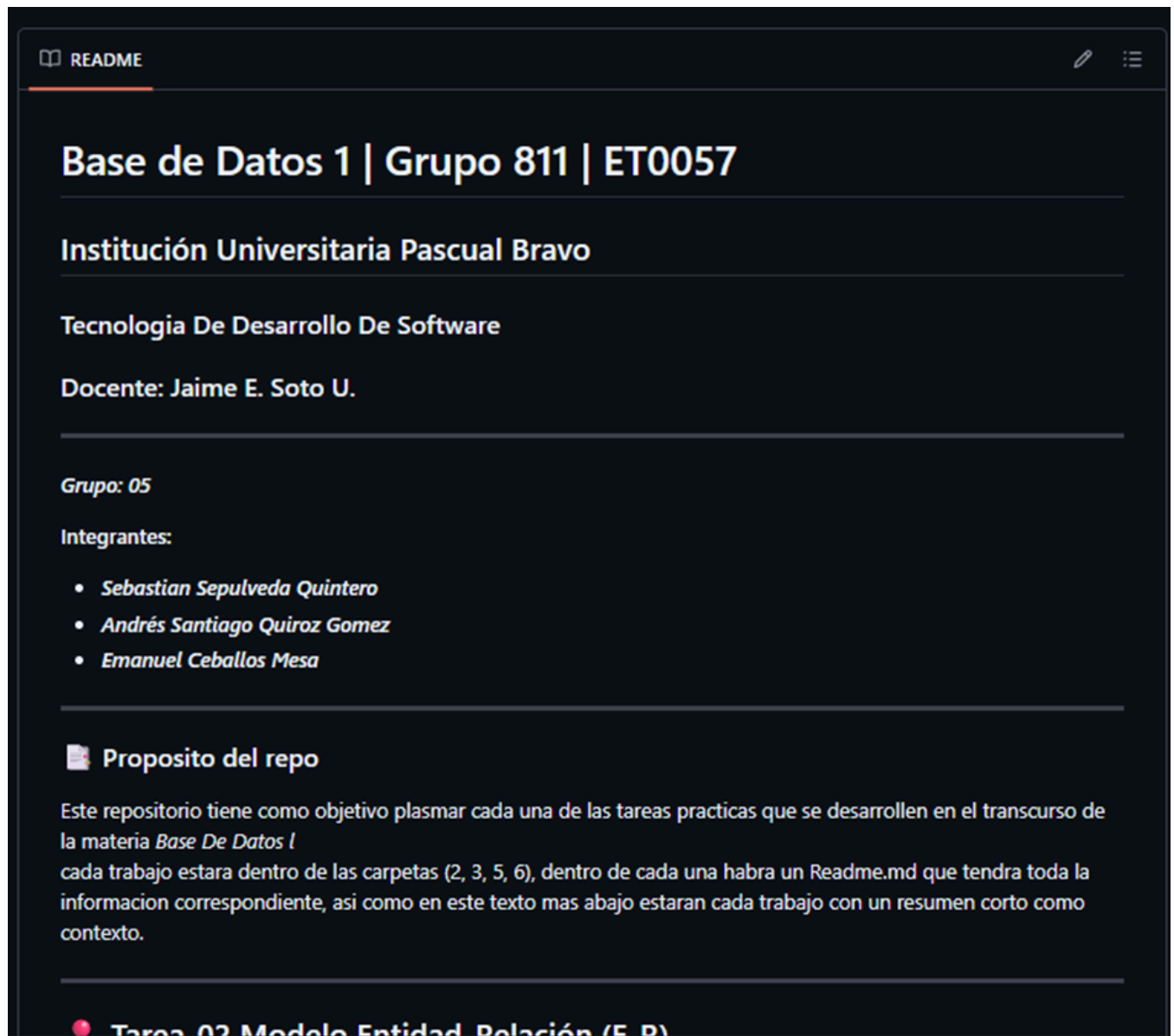
Cuando pasé a las consultas avanzadas con múltiples JOIN fue donde todo se complicó bastante. Hacer un JOIN entre 2 tablas está bien, pero cuando llegas a 5 o 6 JOIN conectando pacientes, médicos, especialidades, hospitales y diagnósticos, ahí sí tocaba pensar bien qué estaba haciendo. Me equivoqué varias veces al principio porque se me olvidaba alguna relación o ponía mal las condiciones del ON. Pero después de practicar, entendí que estos JOIN son súper poderosos porque te dejan sacar información completa de toda la base de datos en una sola consulta. Eso es justo lo que se necesita en un hospital: ver todo el contexto de un paciente sin tener que buscar en 10 tablas diferentes.

Si tengo que decir qué fue lo más importante que aprendí, diría que es tener respeto por los datos. Antes veía las bases de datos como algo técnico y frío, pero ahora entiendo que son la memoria de una organización. Cada INSERT, UPDATE o DELETE tiene consecuencias. Por eso hay que hacerlo con cuidado, validando todo, usando transacciones, y pensando en qué puede salir mal. Las propiedades ACID no son solo teoría, son la garantía de que los datos van a estar seguros y correctos.

Este proyecto me dejó listo para enfrentar cosas más complejas. Ahora puedo sentarme frente a cualquier base de datos y no solo hacer consultas básicas, sino diseñar operaciones complejas, validarlas y asegurarme de que todo funcione bien. Entiendo por qué las empresas buscan gente que sepa de bases de datos: porque los datos son lo más valioso que tienen. Me siento más preparado para el mundo laboral y con ganas de seguir aprendiendo cosas más avanzadas.

9.- Informe

10.- Repositorio



11.- Video de Sustentación

Institución Universitaria Pascual Bravo
Facultad de Ingeniería
Departamento de Sistemas Digitales

The screenshot shows the pgAdmin 4 interface. On the left, the Object Explorer displays a database structure with tables like 'asignacion', 'ciudad_pais', 'cuarto', 'diagnostico', 'diagnostico_medico', 'diagnostico_paciente', 'enfermera', 'enfermera_paciente', 'eps', 'especialidad', 'hospital', 'hospitalizaciones', 'medicamento', 'medico', 'medico_especialidad', 'medico_paciente', 'paciente', 'planta', 'tarjeta_visita', 'tratamiento', 'tratamiento_medicamento', 'tratamiento_paciente', and 'visita'. The main pane shows a query history with the following SQL queries:

```
21 update paciente
22 set email = 'levyatto@gmail.com'
23 where id_paciente = 10
24 --El paciente reportó que el email registrado era de su antiguo trabajo.Solicitó actual
25
26 UPDATE paciente
27 SET ciudad_id = (SELECT ciudad_id FROM ciudad_pais WHERE nombre_ciudad = 'Envigado')
28 WHERE id_paciente = 23;
29 --La paciente se mudó de Medellín a Envigado por motivos laborales.
30
31 UPDATE paciente
32 SET eps_id = (SELECT eps_id FROM eps WHERE nombre_eps = 'Surá')
33 WHERE id_paciente = 35
34 --El paciente cambió de empleador y su nueva empresa tiene convenio con otra EPS.
35
36 UPDATE paciente
37 SET hospital_id = (SELECT hospital_id FROM hospital WHERE nombre_hospital = 'Hospital Pablo Tobón Uribe')
38 WHERE id_paciente = 60;
39 --La solicitó ser atendida en el Hospital Pablo Tobón Uribe porque su médico especialista trabaja allí.
40
41
```

The Data Output pane shows a message: "No data output. Execute a query to get output." A video call window is visible in the top right corner, showing a person with long dark hair wearing a black t-shirt.

The screenshot shows a Bandicam recording of a SQL script. The script is titled "20252-PA-et0057-tia6-DML-equipo-5-02-03-scripts-m" and contains the following SQL queries:

```
--
-- Tarea 6 Parte 12 del Proyecto de Asis
-- SCRIPTS DE MODIFICACIÓN DE LA BASE DE DATOS (UPDATE, INSERTS)
--
-- Miembros del grupo
-- Andres Santiago Quintero Gomez
-- Sebastian Sepulveda Quintero
-- Emanuel Coballos Mesa
--
-- INSTRUCCIONES DE MODIFICACIÓN SOLICITADAS
--
-- Instrucciones UPDATE
--
update paciente
set cell = '3104034458'
where id_paciente = 5;
--el paciente cambio de operador y cambió su línea telefónica. Es necesario actualizar su número de contacto

update paciente
set email = 'levyatto@gmail.com'
where id_paciente = 10
--El paciente reportó que el email registrado era de su antiguo trabajo.Solicitó actualizar a su email personal

UPDATE paciente
SET ciudad_id = (SELECT ciudad_id FROM ciudad_pais WHERE nombre_ciudad = 'Envigado')
WHERE id_paciente = 23;
--La paciente se mudó de Medellín a Envigado por motivos laborales.

UPDATE paciente
SET eps_id = (SELECT eps_id FROM eps WHERE nombre_eps = 'Surá')
WHERE id_paciente = 35
--El paciente cambió de empleador y su nueva empresa tiene convenio con otra EPS.
```

The Bandicam interface shows the script in the main pane, a video call window in the top right corner, and a status bar at the bottom indicating 100% zoom and UTF-8 encoding.

Institución Universitaria Pascual Bravo
Facultad de Ingeniería
Departamento de Sistemas Digitales

RÚBRICA (TIA5)

Tarea Manipulación de Base de Datos Física

Las instrucciones y criterios de cada ítem se encuentran en el ítem en letras azules itálicas

#	Ítems Tarea	Peso	Cal
1	Poblamiento de la Base de Datos (INSERT). Estrategia de Poblamiento	15	
2	Actualizaciones de la Base de Datos (UPDATE)	5	
3	Actualizaciones de la Base de Datos (DELETE)	5	
4	Creación y uso de vista especial	5	
5	Consultas básicas	10	
6	Consultas avanzadas	15	
7	Validación ACID de la Base de Datos. NOTA: Explicar resultado de test ACID en cada consulta	10	
8	Conclusiones individuales. Nota: Explicar el impacto que ha tenido en su desarrollo académico y profesional. Cada estudiante debe estar bien identificado en sus conclusiones (300 palabras mínimo)	5	
9	Informe. Calidad de entrega, se utiliza la plantilla adecuadamente, realiza la entrega de los productos en los archivos asignados, calidad de presentación de los resultados.	5	
10	Repositorio GIT. Nota: Debe estar bien identificado y con la estructura en carpetas solicitada desde el inicio del semestre.	5	
11	Video de Sustentación. Nota: ES OBLIGATORIO. Se evalúa la calidad del Video. Cada participante se presenta adecuadamente con su nombre y con su rostro. Cada uno debe explicar su participación y mostrar código en ejecución en el SGBD. Si no se presenta el Video, la tarea se evaluará sobre 3 puntos como máxima nota.	20	
	NOTA = xx/100 =	Total	100

Nota: Eliminar todas las instrucciones en azul. Solo resultados