

CODING BOOTCAMPS ESPOL

Modulo 2: Database SQL and query optimization

Proyecto Integrador del Módulo

Integrantes

RODAS CRUZ MIA FIORELLA

RODRIGUEZ BUSTAMANTE EMILY ALEJANDRA

TINGO BORBOR CARLOS RAUL

UGALDE ALMEIDA KATTY LORENA

VILLACÍS MORÁN CRISTINA BELÉN

ALLISON BRIGETTE VINUEZA GUTIERREZ

Profesora:

Nicole Agila

Mentora:

Anabella Sanchez

2025 - 2026

Objetivo:

En base al modelo base que se presentó, ajustar e implementar un modelo de base de datos normalizado (hasta 3FN) que permita almacenar y analizar la información de uno de los tres proyectos entregados, demostrando competencias en modelado de datos, normalización y programación SQL.

Objetivos específicos:

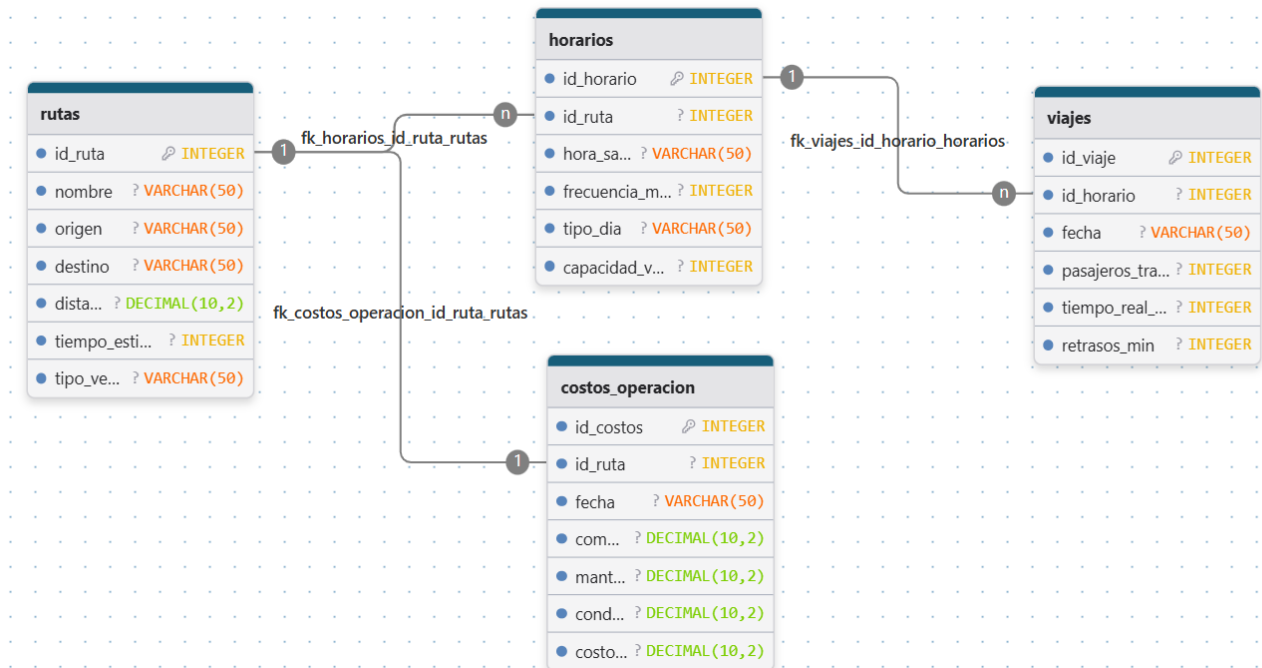
- Diseñar una base de datos para registrar datos de rutas, horarios y uso del transporte urbano.
- Generar reportes sobre patrones de uso y eficiencia de las rutas.
- Identificar oportunidades para optimizar la movilidad urbana.

¿De qué trata el proyecto?

El objetivo de nuestro proyecto fue diseñar y optimizar una base de datos para analizar las redes de transporte urbano. Buscamos mejorar la eficiencia de las rutas y horarios para reducir los tiempos de viaje y la congestión, basándonos en datos reales. Para lograr esto, creamos una base de datos que permite registrar, analizar y generar reportes.

1. Analizar los datos CSV para identificar entidades, atributos y relaciones

En base a los CSV correspondientes se pudo identificar 5 entidades que son rutas, horarios, costos_operacion y viajes con tipos de datos como integer, varchar y decimal.



2. Aplicar las reglas de normalización hasta Tercera Forma Normal (3FN)

1FN

El modelo cumple con 1FN que asegura atomicidad en cada celda.

2FN

El modelo cumple con 2FN ya que no tiene dependencias parciales.

3FN

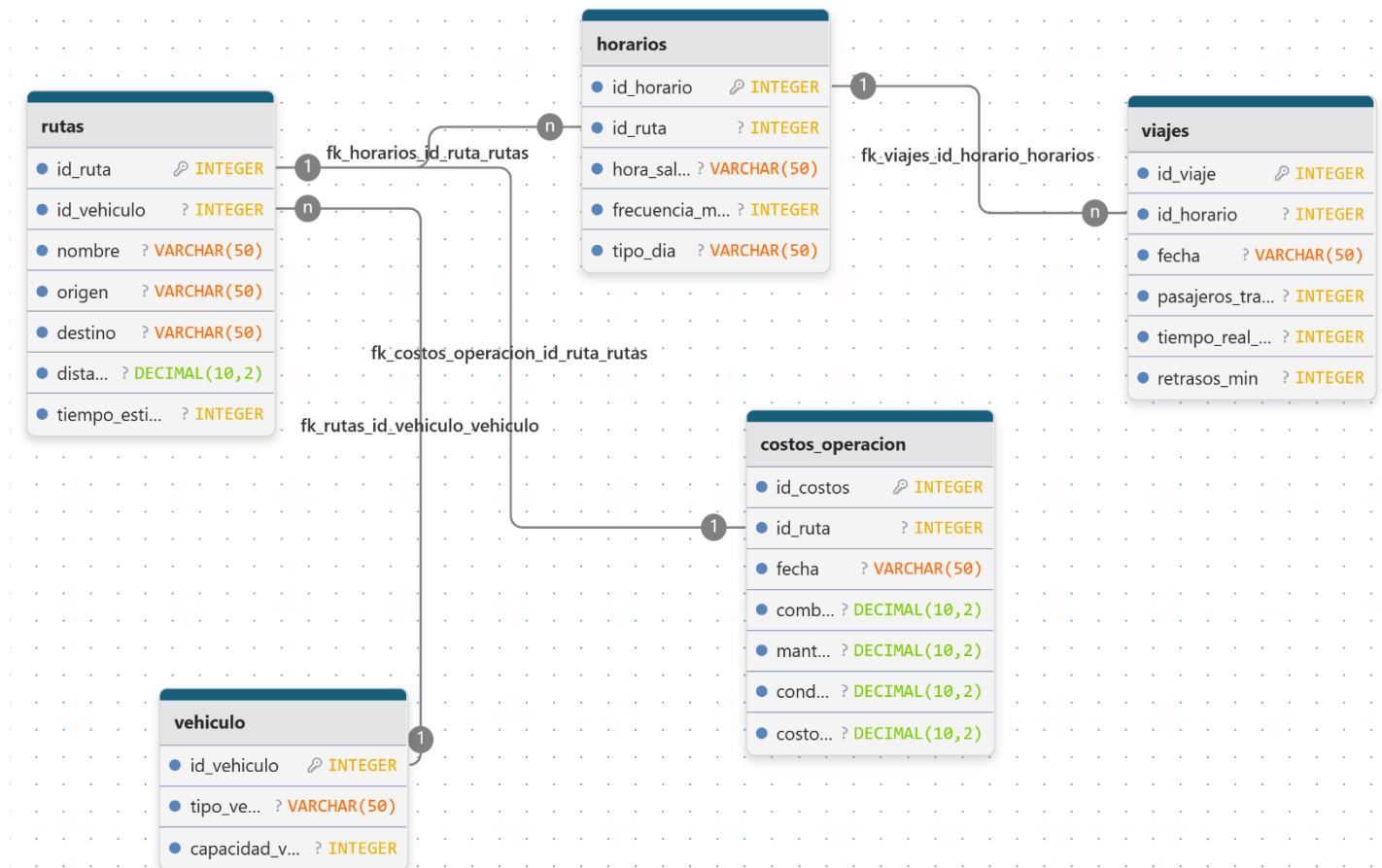
El modelo presenta una dependencia transitiva en la tabla de horarios y rutas; en horarios la columna capacidad_vehiculo que depende de id_ruta y no a su PK id_horario en el caso de la tabla de rutas la columna tipo_vehiculo no tendría mucha relacion por lo cual se recomienda agruparla en una nueva tabla llamada vehiculo.



3. Diseñar un modelo relacional eficiente y sin redundancias

El diseño evita duplicación de información, ya que cada tabla concentra un conjunto específico de datos (vehículos, rutas, horarios, viajes, costos).

Las relaciones establecidas mediante claves primarias y foráneas garantizan la consistencia y trazabilidad, facilitando consultas complejas como ocupación de rutas, retrasos promedio y costo por pasajero.



7. Crear scripts SQL para definición de base de datos y tablas

G8_ScriptNormalizado.sql: Script base del modelo ya normalizado a 3FN.

Diseño e implementacion de base de datos.sql: Script de ejecución donde se realizó carga de datos, inserts y consultas.

8. Cargar datos utilizando múltiples métodos (INSERT y Load Data Wizard)

- Al menos para una tabla implementar código de Insert.

The screenshot displays a database management interface. On the left, the 'Navigator' pane shows a tree structure of schemas, with 'transporte_urbano' selected. The main area shows a SQL query editor with the following code:

```
1 CREATE DATABASE transporte_urbano;
2
3 use transporte_urbano;
4
5 SELECT * FROM rutas;
6 SELECT * FROM
```

Below the editor, the 'Result Grid' shows the output of the query, displaying 10 rows of data for the 'rutas' table. The columns are: id_ruta, nombre, origen, destino, distancia_km, tiempo_estimado_min, and tipo_vehiculo.

	id_ruta	nombre	origen	destino	distancia_km	tiempo_estimado_min	tipo_vehiculo
▶	1	Ruta Norte	Terminal Norte	Centro Histórico	12.5	35	Autobús
	2	Ruta Sur	Terminal Sur	Plaza Principal	18.2	45	Articulado
	3	Ruta Este	Estación Este	Universidad	8.7	25	Minibús
	4	Ruta Oeste	Terminal Oeste	Hospital General	15.3	40	Autobús
	5	Ruta Centro	Plaza Central	Aeropuerto	22.1	55	Articulado
	6	Ruta Periférico	Zona Industrial	Centro Comercial	25.8	60	Articulado
	7	Ruta Residencial	Colonia Jardines	Metro Centro	11.4	30	Minibús
	8	Ruta Express	Terminal Central	Zona Financiera	16.9	38	Autobús
	9	Ruta Universitaria	Campus Norte	Campus Sur	13.6	32	Minibús
	10	Ruta Turística	Hotel Zone	Museo Nacional	9.8	28	Minibús

At the bottom, the 'Output' pane shows the execution log for the query, indicating that the data was successfully loaded into the 'transporte_urbano' database.

#	Time	Action	Message
✓ 8	00:14:41	PREPARE stmt FROM 'INSERT INTO `transporte_urbano`.`rutas` (`id_ruta`,`nombre`,`origen`,...	OK
✓ 9	00:14:42	DEALLOCATE PREPARE stmt	OK
✓ 10	00:14:48	SELECT * FROM rutas LIMIT 0, 1000	10 row(s) returned

Navigator: SCHEMAS

Filter objects

- restaurante_gourmet
- restaurante_join
- restaurante_practicas
- sesion7_sql
- sesión3
- sys
- taller_sql
- transporte_urbano**
 - Tables
 - horarios
 - rutas
 - Views
 - Stored Procedures

Administration Schemas

Information

Schema: **transporte_urbano**

Query 1

```

1 CREATE DATABASE transporte_urbano;
2
3 use transporte_urbano;
4
5 SELECT * FROM rutas;
6 SELECT * FROM horarios;

```

Result Grid

	id_horario	id_ruta	hora_salida	frecuencia_min	tipo_dia	capacidad_vehiculo
1	1	1	05:30:00	15	Laboral	40
2	1	1	06:00:00	10	Laboral	40
3	1	1	06:30:00	10	Laboral	40
4	1	1	07:00:00	8	Laboral	40
5	1	1	07:30:00	8	Laboral	40
6	1	1	08:00:00	8	Laboral	40
7	1	1	08:30:00	10	Laboral	40
8	1	1	09:00:00	15	Laboral	40
9	1	1	17:00:00	8	Laboral	40
10	1	1	17:30:00	8	Laboral	40
11	1	1	18:00:00	8	Laboral	40
12	1	1	18:30:00	10	Laboral	40

horarios 2

Output

Action Output

#	Time	Action	Message
16	00:21:45	PREPARE stmt FROM 'INSERT INTO `transporte_urbano`.`horarios` (`id_horario`,`id_ruta`,`h...	OK
17	00:21:45	DEALLOCATE PREPARE stmt	OK
18	00:21:56	SELECT * FROM horarios LIMIT 0, 1000	77 row(s) returned

Object Info Session

Navigator: SCHEMAS

Filter objects

- restaurante_gourmet
- restaurante_join
- restaurante_practicas
- sesion7_sql
- sesión3
- sys
- taller_sql
- transporte_urbano**
 - Tables
 - horarios
 - rutas
 - viajes
 - Views

Administration Schemas

Information

Schema: **transporte_urbano**

Query 1

```

2
3 use transporte_urbano;
4
5 SELECT * FROM rutas;
6 SELECT * FROM horarios;
7 SELECT * FROM viajes;

```

Result Grid

	id_viaje	id_horario	fecha	pasajeros_transportados	tiempo_real_min	retrasos_min
1	1	1	2024-01-02	25	37	2
2	2	2	2024-01-02	38	36	1
3	3	3	2024-01-02	39	38	3
4	4	4	2024-01-02	40	37	2
5	5	5	2024-01-02	38	39	4
6	6	6	2024-01-02	35	36	1
7	7	7	2024-01-02	32	38	3
8	8	8	2024-01-02	28	36	1
9	9	9	2024-01-02	37	40	2
10	10	10	2024-01-02	39	41	3
11	11	11	2024-01-02	40	42	4
12	12	12	2024-01-02	35	38	3

viajes 3

Output

Action Output

#	Time	Action	Message
24	00:25:39	PREPARE stmt FROM 'INSERT INTO `transporte_urbano`.`viajes` (`id_viaje`,`id_horario`,`fec...	OK
25	00:25:40	DEALLOCATE PREPARE stmt	OK
26	00:26:00	SELECT * FROM viajes LIMIT 0, 1000	200 row(s) returned

Object Info Session

Navigator

SCHEMAS

Filter objects

- sesion7_sql
- sesión3
- sys
- taller_sql
- transporte_urbano**
 - costos_operacion
 - horarios
 - rutas
 - viajes
- Views
- Stored Procedures
- Functions

Administration Schemas

Information

Schema:
transporte_urbano

Query 1

```

3 • use transporte_urbano;
4
5 • SELECT * FROM rutas;
6 • SELECT * FROM horarios;
7 • SELECT * FROM viajes;
8 • SELECT * FROM costos_operacion;

```

Result Grid

	id_costo	id_ruta	fecha	combustible	mantenimiento	conductor	costo_total
1	1	1	2024-01-02	245.5	85	320	650.5
2	2	2	2024-01-02	380.75	120	340	840.75
3	3	3	2024-01-02	165.3	65	280	510.3
4	4	4	2024-01-02	285.4	95	320	700.4
5	5	5	2024-01-02	420.6	140	360	920.6
6	6	6	2024-01-02	475.8	160	380	1015.8
7	7	7	2024-01-02	195.25	70	290	555.25
8	8	8	2024-01-02	315.45	105	330	750.45
9	9	9	2024-01-02	225.35	80	300	605.35
10	10	10	2024-01-02	178.9	60	270	508.9
11	1	1	2024-01-03	248.2	85	320	653.2
12	2	2	2024-01-03	385.4	120	340	845.4

costos_operacion 4 x

Output

Action Output

#	Time	Action	Message
32	00:27:22	PREPARE stmt FROM 'INSERT INTO `transporte_urbano`.`costos_operacion` (`id_costo`,`id...	OK
33	00:27:22	DEALLOCATE PREPARE stmt	OK
34	00:27:28	SELECT * FROM costos_operacion LIMIT 0, 1000	50 row(s) returned

Object Info Session

Navigator

SCHEMAS

Filter objects

- restaurante_practicas
- sesion7_sql
- sesión3
- sys
- taller_sql
- transporte_urbano**
 - costos_operacion
 - horarios
 - rutas
 - vehiculos
 - viajes
- Views

Administration Schemas

Information

Schema:
transporte_urbano

Query 1

```

4
5 • SELECT * FROM rutas;
6 • SELECT * FROM horarios;
7 • SELECT * FROM viajes;
8 • SELECT * FROM costos_operacion;
9 • SELECT * FROM vehiculos;
10
11 • CREATE TABLE Vehiculos (
12     tipo_vehiculo VARCHAR(50) PRIMARY KEY,
13     capacidad INT NOT NULL
14 );
15

```

Result Grid

	tipo_vehiculo	capacidad
*	NULL	NULL

vehiculos 5 x

Output

Action Output

#	Time	Action	Message
34	00:57:43	SELECT * FROM costos_operacion LIMIT 0, 1000	50 row(s) returned
35	01:14:42	CREATE TABLE Vehiculos (tipo_vehiculo VARCHAR(50) PRIMARY KEY, capacidad I...	0 row(s) affected
36	01:14:53	SELECT * FROM vehiculos LIMIT 0, 1000	0 row(s) returned

Object Info Session

Navigator: SCHEMAS

Filter objects

- restaurante_practicas
- sesion7_sql
- sesión3
- sys
- taller_sql
- transporte_urbano**
 - Tables
 - costos_operacion
 - horarios
 - rutas
 - vehiculos
 - viajes

Administration Schemas

Information: Schema: **transporte_urbano**

Query 1

```

10
11 CREATE TABLE Vehiculos (
12     tipo_vehiculo VARCHAR(50) PRIMARY KEY,
13     capacidad INT NOT NULL
14 );
15
16 INSERT INTO Vehiculos (tipo_vehiculo, capacidad) VALUES ('Autobús', 40);
17 INSERT INTO Vehiculos (tipo_vehiculo, capacidad) VALUES ('Articulado', 80);
18 INSERT INTO Vehiculos (tipo_vehiculo, capacidad) VALUES ('Minibús', 25);
19
20

```

Result Grid

tipo_vehiculo	capacidad
Articulado	80
Autobús	40
Minibús	25
NULL	NULL

vehiculos 6 x

Output

Action Output

#	Time	Action	Message
38	01:18:54	INSERT INTO Vehiculos (tipo_vehiculo, capacidad) VALUES ('Articulado', 80)	1 row(s) affected
39	01:18:54	INSERT INTO Vehiculos (tipo_vehiculo, capacidad) VALUES ('Minibús', 25)	1 row(s) affected
40	01:19:00	SELECT * FROM vehiculos LIMIT 0, 1000	3 row(s) returned

Object Info Session

Query Completed

Navigator: SCHEMAS

Filter objects

- sys
- taller_sql
- transporte_urbano**
 - Tables
 - costos_operacion
 - horarios
 - rutas
 - vehiculos
 - viajes
 - Views
 - Stored Procedures
 - Functions

Administration Schemas

Information: No object selected

Diseño e implementación de bas...

```

23 ALTER TABLE Rutas
24     MODIFY COLUMN nombre VARCHAR(50) NOT NULL,
25     MODIFY COLUMN origen VARCHAR(50) NOT NULL,
26     MODIFY COLUMN destino VARCHAR(50) NOT NULL,
27     MODIFY COLUMN tipo_vehiculo VARCHAR(50) NOT NULL,
28     MODIFY COLUMN distancia_km DECIMAL(5, 2) NOT NULL,
29     MODIFY COLUMN tiempo_estimado_min INT NOT NULL;
30
31 ALTER TABLE Rutas
32     ADD FOREIGN KEY (tipo_vehiculo) REFERENCES Vehiculos(tipo_vehiculo);

```

Result Grid

id_ruta	nombre	origen	destino	distancia_km	tiempo_estimado_min	tipo_vehiculo
1	Ruta Norte	Terminal Norte	Centro Histórico	12.50	35	Autobús
2	Ruta Sur	Terminal Sur	Plaza Principal	18.20	45	Articulado
3	Ruta Este	Estación Este	Universidad	8.70	25	Minibús
4	Ruta Oeste	Terminal Oeste	Hospital General	15.30	40	Autobús
5	Ruta Centro	Plaza Central	Aeropuerto	22.10	55	Articulado
6	Ruta Periférico	Zona Industrial	Centro Comercial	25.80	60	Articulado
7	Ruta Residencial	Colonia Jardines	Metro Centro	11.40	30	Minibús
8	Ruta Europea	Terminal Central	Zona Financiera	16.00	30	Autobús

rutas 9 x

Output

Action Output

#	Time	Action	Message
26	11:43:09	SELECT * FROM rutas LIMIT 0, 1000	10 row(s) returned
27	11:43:32	SELECT * FROM vehiculos LIMIT 0, 1000	3 row(s) returned
28	11:43:36	SELECT * FROM rutas LIMIT 0, 1000	10 row(s) returned

Object Info Session

Navigator: Diseño e implementación de bas... x

SCHEMAS

Filter objects

- sys
- taller_sql
- transporte_urbano
 - Tables
 - costos_operacion
 - horarios
 - rutas
 - vehiculos
 - viajes
 - Views
 - Stored Procedures
 - Functions

Administration Schemas

Information

No object selected

Result Grid

Filter Rows: Edit: Export/Import: Wrap Cell Content:

	id_horario	id_ruta	hora_salida	frecuencia_min	tipo_dia	capacidad_vehiculo
1	1	1	05:30:00	15	Laboral	40
2	1	1	06:00:00	10	Laboral	40
3	1	1	06:30:00	10	Laboral	40
4	1	1	07:00:00	8	Laboral	40
5	1	1	07:30:00	8	Laboral	40
6	1	1	08:00:00	8	Laboral	40
7	1	1	08:30:00	10	Laboral	40
8	1	1	09:00:00	15	Laboral	40

horarios 11 x Apply Revert

Output

Action Output

#	Time	Action	Message
31	11:57:17	ALTER TABLE Horarios MODIFY COLUMN frecuencia_min INT NOT NULL, MODIFY COL...	77 row(s) affected Records: 77 Duplicates:
32	11:57:18	ALTER TABLE Horarios ADD FOREIGN KEY (id_ruta) REFERENCES Rutas(id_ruta)	77 row(s) affected Records: 77 Duplicates:
33	11:57:26	SELECT * FROM horarios LIMIT 0, 1000	77 row(s) returned

Object Info Session

Navigator: Diseño e implementación de bas... x

SCHEMAS

Filter objects

- sys
- taller_sql
- transporte_urbano
 - Tables
 - costos_operacion
 - horarios
 - rutas
 - vehiculos
 - viajes
 - Views
 - Stored Procedures
 - Functions

Administration Schemas

Information

No object selected

Result Grid

Filter Rows: Edit: Export/Import: Wrap Cell Content:

	id_viaje	id_horario	fecha	pasajeros_transportados	tiempo_real_min	retrasos_min
1	1	1	2024-01-02	25	37	2
2	2	2	2024-01-02	38	36	1
3	3	3	2024-01-02	39	38	3
4	4	4	2024-01-02	40	37	2
5	5	5	2024-01-02	38	39	4
6	6	6	2024-01-02	35	36	1
7	7	7	2024-01-02	32	38	3
8	8	8	2024-01-02	30	36	1

viajes 13 x Apply Revert

Output

Action Output

#	Time	Action	Message
36	12:04:36	ALTER TABLE Viajes MODIFY COLUMN fecha DATE NOT NULL, MODIFY COLUMN pasa...	200 row(s) affected Records: 200 Duplicates:
37	12:04:37	ALTER TABLE Viajes ADD FOREIGN KEY (id_horario) REFERENCES Horarios(id_horario)	200 row(s) affected Records: 200 Duplicates:
38	12:04:45	SELECT * FROM viajes LIMIT 0, 1000	200 row(s) returned

Object Info Session

Navigator

SCHEMAS

Filter objects

sys

taller_sql

transporte_urbano

Tables

costos_operacion

horarios

rutas

vehiculos

viajes

Views

Stored Procedures

Functions

Administration

Schemas

Information

No object selected

Object Info

Session

Diseño e implementación de bas...

Limit to 1000 rows

63

ALTER TABLE costos_Operacion

64

MODIFY COLUMN fecha DATE NOT NULL,

65

MODIFY COLUMN combustible DECIMAL(6, 2) NOT NULL,

66

MODIFY COLUMN mantenimiento INT NOT NULL,

67

MODIFY COLUMN conductor INT NOT NULL,

68

MODIFY COLUMN costo_total DECIMAL(7, 2) NOT NULL;

69

70

ALTER TABLE costos_Operacion

71

ADD FOREIGN KEY (id_ruta) REFERENCES Rutas(id_ruta);

72

Result Grid

Filter Rows:

Edit

Export/Import

Wrap Cell Content:

Result Grid

Form Editor

	id_costo	id_ruta	fecha	combustible	mantenimiento	conductor	costo_total
▶	1	1	2024-01-02	245.50	85	320	650.50
	2	2	2024-01-02	380.75	120	340	840.75
	3	3	2024-01-02	165.30	65	280	510.30
	4	4	2024-01-02	285.40	95	320	700.40
	5	5	2024-01-02	420.60	140	360	920.60
	6	6	2024-01-02	475.80	160	380	1015.80
	7	7	2024-01-02	195.25	70	290	555.25
	8	8	2024-01-02	215.45	105	320	750.45

costos_operacion 16 x


Apply

Revert

Output

Action Output

#	Time	Action	Message
✓ 44	12:13:00	ALTER TABLE costos_Operacion MODIFY COLUMN fecha DATE NOT NULL, MODIFY C...	50 row(s) affected Records: 50 Duplicates:
✓ 45	12:13:05	ALTER TABLE costos_Operacion MODIFY COLUMN fecha DATE NOT NULL, MODIFY C...	0 row(s) affected Records: 0 Duplicates: C
✓ 46	12:13:12	SELECT * FROM costos_operacion LIMIT 0, 1000	50 row(s) returned

CODING
BOOTCAMPS
espol

Planificación urbana – Horas pico

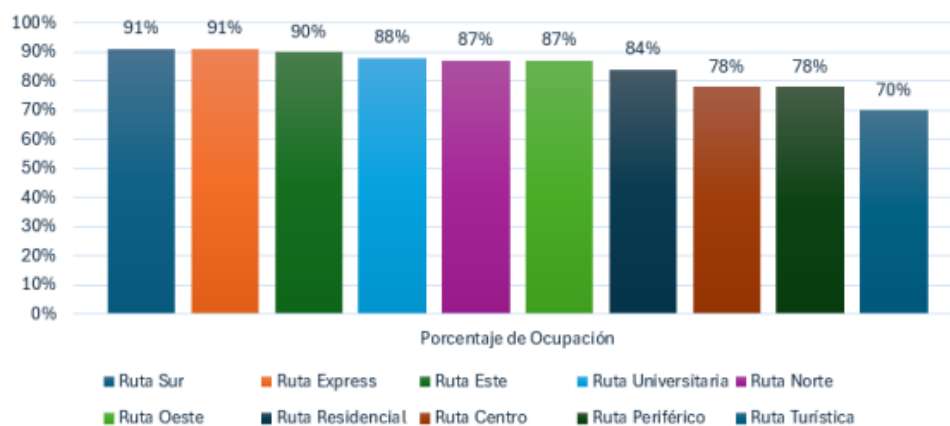
Los resultados muestran que las rutas con mayor ocupación en horas pico son Ruta Sur (91%), Ruta Express (91%) y Ruta Este (90%), seguidas por la Ruta Universitaria (88%) y la Ruta Norte (87%). En contraste, Ruta Turística (70%), Centro (78%) y Periférico (78%) presentan menor demanda. Esto indica que las primeras requieren mayor prioridad en asignación de capacidad y frecuencias.

```
75 -- Planificación urbana
76 -- Horas pico
77 • SELECT T2.nombre,
78      CONCAT(TRUNCATE(AVG(T1.pasajeros_transportados / T3.capacidad) * 100, 0), '%')
79      AS porcentaje_ocupacion
80 FROM Viajes AS T1
81 JOIN Horarios AS T4 ON T1.id_horario = T4.id_horario
82 JOIN Rutas AS T2 ON T4.id_ruta = T2.id_ruta
83 JOIN Vehiculos AS T3 ON T2.tipo_vehiculo = T3.tipo_vehiculo
84 WHERE TIME(T4.hora_salida) >= '06:00:00' AND TIME(T4.hora_salida) < '09:00:00'
85       OR TIME(T4.hora_salida) >= '17:00:00' AND TIME(T4.hora_salida) < '19:00:00'
86 GROUP BY T2.nombre
87 ORDER BY porcentaje_ocupacion DESC;
```

result Grid | Filter Rows: | Export: | Wrap Cell Content: [IA](#)

nombre	porcentaje_ocupacion
Ruta Sur	91%
Ruta Express	91%
Ruta Este	90%
Ruta Universitaria	88%
Ruta Norte	87%
Ruta Oeste	87%
Ruta Residencial	84%
Ruta Centro	78%
Ruta Periférico	78%
Ruta Turística	70%

Planificación urbana – Horas pico



¿Qué rutas tienen mayor ocupación en horas pico?

La consulta muestra que las rutas Sur, Express y Este superan el 90% de ocupación, evidenciando la necesidad de asignar vehículos de mayor capacidad y frecuencias más cortas en esos tramos.

nombre	tipo_vehiculo	capacidad_vehiculo
Ruta Norte	Autobús	40
Ruta Sur	Articulado	80
Ruta Este	Minibús	25
Ruta Oeste	Autobús	40
Ruta Centro	Articulado	80
Ruta Periférico	Articulado	80
Ruta Residencial	Minibús	25
Ruta Express	Autobús	40
Ruta Universitaria	Minibús	25
Ruta Turística	Minibús	25

Gestión temporal – Horarios menor uso

Con respecto a la consulta los horarios con menor utilización del transporte se presentan durante los fines de semana, especialmente a las 10:15 con 10,5 pasajeros en promedio, 09:30 con 12,5 y 09:45 con 14,0. Estos resultados evidencian una demanda reducida en dichas franjas, lo que sugiere la posibilidad de ajustar la frecuencia de los servicios para optimizar la asignación de recursos.

```

89 -- Gestión temporal
90 -- Horarios con menor uso de transporte
91 • SELECT h.hora_salida,
92        h.tipo_dia,
93        ROUND(AVG(v.pasajeros_transportados),2) AS promedio_pasajeros
94 FROM Viajes v
95 JOIN Horarios h ON v.id_horario = h.id_horario
96 GROUP BY h.hora_salida, h.tipo_dia
97 ORDER BY promedio_pasajeros ASC;

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

hora_salida	tipo_dia	promedio_pasajeros
10:15:00	Fin_semana	10.50
09:30:00	Fin_semana	12.50
09:45:00	Fin_semana	14.00
13:00:00	Fin_semana	15.00
08:00:00	Fin_semana	15.50
10:00:00	Fin_semana	15.60
16:00:00	Laboral	18.50
12:00:00	Laboral	19.50
09:15:00	Fin_semana	24.50
09:00:00	Fin_semana	25.67
09:00:00	Laboral	28.00
08:30:00	Fin_semana	28.50
05:30:00	Laboral	28.50
08:30:00	Laboral	32.00
10:30:00	Laboral	34.00

Result 12 x

¿Qué horarios presentan menor uso del transporte?

Los resultados señalan que, durante los fines de semana, especialmente entre las 09:30 y 10:15, la demanda promedio es la más baja (entre 10 y 15 pasajeros). Esto sugiere que en esas franjas horarias podría reducirse la frecuencia de viajes para optimizar recursos.

Hora	Tipo de dia	Promedio de Pasajeros
10:15:00	Fin semana	10,5
9:30:00	Fin semana	12,5
9:45:00	Fin semana	14,00
13:00:00	Fin semana	15,00
8:00:00	Fin semana	15,50
10:00:00	Fin semana	15,60
16:00:00	Laboral	18,50
12:00:00	Laboral	19,50
9:15:00	Fin semana	24,50

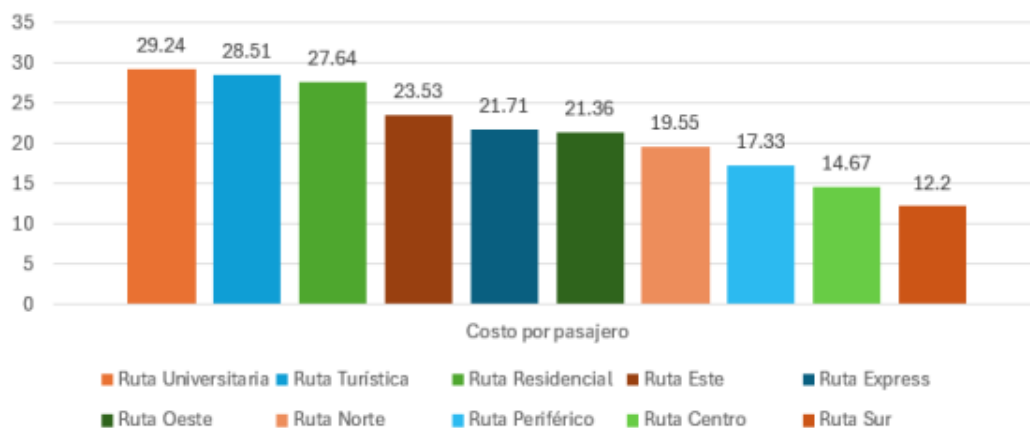
Métricas – Costo por pasajero

Una métrica clave para evaluar la optimización de las rutas es el costo por pasajero. Los resultados indican mayores valores en rutas como la Universitaria (29,24), Turística (28,51) y Residencial (27,64), reflejando menor eficiencia, mientras que la Sur (12,20) y la Periférico (16,70) son más rentables. A este indicador se suman el tiempo promedio de viaje y el retraso promedio, que permiten analizar eficiencia y puntualidad operativa.

```
102  -- Métricas
103  -- Costo Por Pasajero
104  • SELECT T2.nombre,
105     TRUNCATE(SUM(T1.costo_total) / SUM(T4.pasajeros_transportados), 2) AS costo_por_pasajero
106  FROM costos_operacion AS T1
107  INNER JOIN rutas AS T2 ON T1.id_ruta = T2.id_ruta
108  INNER JOIN horarios AS T3 ON T2.id_ruta = T3.id_ruta
109  INNER JOIN viajes AS T4 ON T3.id_horario = T4.id_horario AND T1.fecha = T4.fecha
110  GROUP BY T2.nombre
111  ORDER BY costo_por_pasajero DESC;
```

nombre	costo_por_pasajero
Ruta Universitaria	29.24
Ruta Turística	28.51
Ruta Residencial	27.64
Ruta Este	23.53
Ruta Express	21.71
Ruta Oeste	21.36
Ruta Norte	19.55
Ruta Periférico	17.33
Ruta Centro	14.67
Ruta Sur	12.20

Métricas – Costo por pasajero



¿Qué métricas pueden usarse para evaluar la optimización de las rutas?

Usando la métrica costo por pasajero las rutas como la Universitaria y Turística presentan los valores más altos, indicando menor eficiencia.

Función ventana

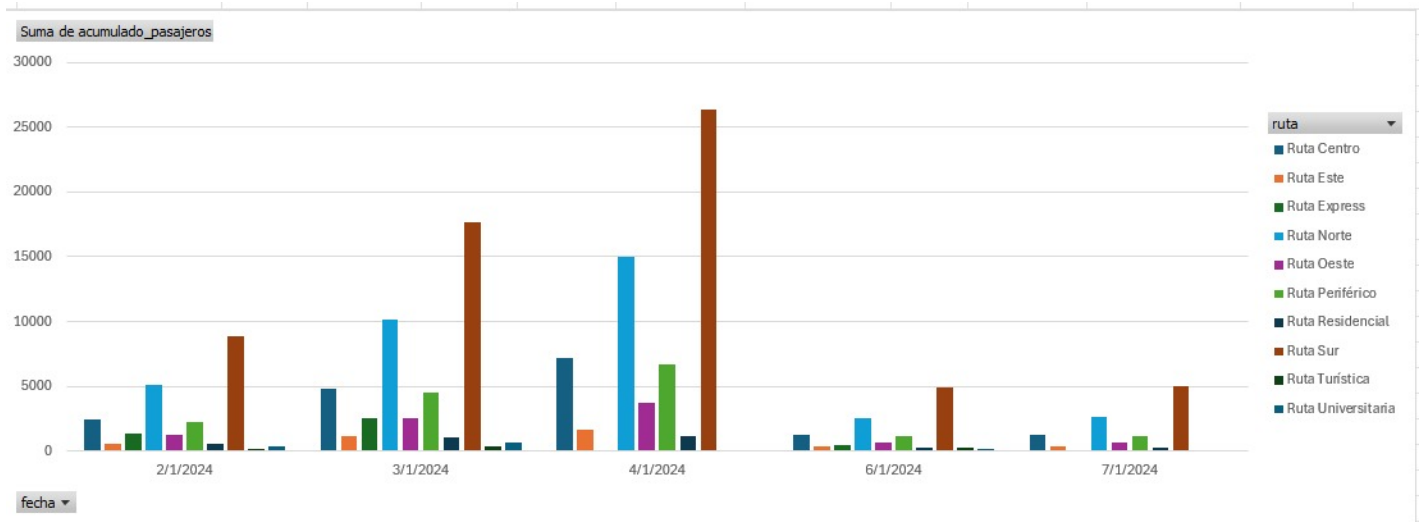
Esta consulta utiliza una función ventana con SUM() OVER para calcular el acumulado de pasajeros transportados en cada ruta a lo largo del tiempo, sin perder el detalle por fecha.

```
113 -- FUNCION VENTANA
114 -- Evolución de pasajeros acumulados por ruta
115 -- Esta consulta muestra cómo se acumulan los pasajeros transportados
116 -- en cada ruta a lo largo del tiempo (fecha).
117 -- Permite analizar la tendencia de crecimiento y detectar picos de demanda.
118
119 • SELECT r.nombre AS ruta,
120        v.fecha,
121        v.pasajeros_transportados,
122        SUM(v.pasajeros_transportados)
123          OVER (PARTITION BY r.nombre ORDER BY v.fecha) AS acumulado_pasajeros
124 FROM Viajes v
125 JOIN Horarios h ON v.id_horario = h.id_horario
126 JOIN Rutas r ON h.id_ruta = r.id_ruta
127 ORDER BY r.nombre, v.fecha;
```

Result Grid				
Filter Rows:		Export: Wrap Cell Content: IA		
	ruta	fecha	pasajeros_transportados	acumulado_pasajeros
▶	Ruta Centro	2024-01-02	62	403
	Ruta Centro	2024-01-02	66	403
	Ruta Centro	2024-01-02	65	403
	Ruta Centro	2024-01-02	72	403
	Ruta Centro	2024-01-02	68	403
	Ruta Centro	2024-01-02	70	403
	Ruta Centro	2024-01-03	61	800
	Ruta Centro	2024-01-03	67	800
	Ruta Centro	2024-01-03	71	800
	Ruta Centro	2024-01-03	65	800
	Ruta Centro	2024-01-03	69	800
	Ruta Centro	2024-01-03	64	800
	Ruta Centro	2024-01-04	66	1191
	Ruta Centro	2024-01-04	70	1191
	Ruta Centro	2024-01-04	63	1191
	Ruta Centro	2024-01-04	68	1191
	Ruta Centro	2024-01-04	64	1191
	Ruta Centro	2024-01-04	60	1191
	Ruta Centro	2024-01-06	28	1219
	Ruta Centro	2024-01-07	29	1248
	Ruta Este	2024-01-02	25	118

Análisis total acumulado de pasajeros

De esta manera se observa no solo cuántos pasajeros viajaron en un día específico, sino también cómo va creciendo el total acumulado día tras día. Este enfoque es útil porque permite analizar la tendencia de la demanda, identificar picos de crecimiento y evaluar la evolución del servicio por ruta, lo que facilita decisiones sobre asignación de recursos y planificación operativa.



Subconsulta

Esta subconsulta realiza un análisis de puntualidad al calcular el retraso promedio por ruta y compararlo con el promedio global del sistema. Para ello, se obtiene el AVG(retrasos_min) de cada ruta y se aplica un HAVING que filtra únicamente aquellas que superan el retraso promedio de todos los viajes. Los resultados muestran que las rutas Sur (4.00 min), Periférico (3.70 min), Norte (3.63 min) y Centro (3.60 min) presentan mayores retrasos que el promedio global, lo que indica que requieren especial atención en la planificación operativa y posibles ajustes en horarios o frecuencias.

```
135 • SELECT r.nombre,  
136         ROUND(AVG(v.retrasos_min),2) AS retraso_promedio  
137 FROM Viajes v  
138 JOIN Horarios h ON v.id_horario = h.id_horario  
139 JOIN Rutas r ON h.id_ruta = r.id_ruta  
140 GROUP BY r.nombre  
141 HAVING retraso_promedio > (  
142     SELECT ROUND(AVG(retrasos_min),2)  
143     FROM Viajes  
144 );
```

nombre	retraso_promedio
Ruta Norte	3.63
Ruta Sur	4.00
Ruta Centro	3.60
Ruta Periférico	3.70



Optimización de consulta

Se implementó un índice en la columna fecha de la tabla viajes para agilizar las búsquedas y joins por fecha, y un índice compuesto en costos_operacion (id_ruta, fecha) para mejorar el rendimiento en consultas que combinan ruta y fecha. Posteriormente, se utilizó el comando EXPLAIN sobre la consulta de costo por pasajero, lo que permitió verificar que el motor de MySQL aprovecha estos índices, reduciendo la cantidad de filas escaneadas y mejorando la eficiencia en la ejecución. Este proceso asegura que las consultas sean más rápidas y escalables conforme crezca el volumen de datos.

```
151 • CREATE INDEX idx_fecha_viajes ON viajes(fecha);
152
153 -- Índice compuesto en costos_operacion para consultas por ruta y fecha
154 • CREATE INDEX idx_ruta_fecha_costos ON costos_operacion(id_ruta, fecha);
155
156 • EXPLAIN
157 SELECT r.nombre AS ruta,
158        TRUNCATE(SUM(c.costo_total) / NULLIF(SUM(v.pasajeros_transportados),0), 2) AS costo_por_pasajero
159 FROM costos_operacion c
160 JOIN Rutas r ON c.id_ruta = r.id_ruta
161 JOIN Horarios h ON r.id_ruta = h.id_ruta
162 JOIN Viajes v ON h.id_horario = v.id_horario AND c.fecha = v.fecha
163 GROUP BY r.nombre
164 ORDER BY costo_por_pasajero DESC;
```

Result Grid											
Filter Rows: <input type="text"/> Export: Wrap Cell Contents: <input checked="" type="checkbox"/>											
id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	r	NULL	ALL	PRIMARY	NULL	NULL	NULL	10	100.00	Using temporary; Using filesort
1	SIMPLE	h	NULL	ref	PRIMARY, id_ruta	id_ruta	5	transporte_urbano.r.id_ruta	7	100.00	Using index
1	SIMPLE	v	NULL	ref	id_horario, idx_fecha_viajes	id_horario	5	transporte_urbano.h.id_horario	2	100.00	NULL
1	SIMPLE	c	NULL	ref	idx_ruta_fecha_costos	idx_ruta_fecha_costos	8	transporte_urbano.r.id_ruta, transporte_urban...	1	100.00	NULL

Conclusión

El modelo de base de datos y las consultas desarrolladas permitieron analizar de manera integral la operación del sistema de transporte urbano. Los resultados muestran patrones de alta demanda en rutas como Sur, Express y Este, baja utilización en ciertos horarios de fin de semana y diferencias significativas en la eficiencia económica y puntualidad de las rutas. Esta información constituye una herramienta clave para la toma de decisiones, ya que facilita la optimización de frecuencias, la asignación de vehículos adecuados y la mejora en el control de costos, contribuyendo a un servicio más eficiente y orientado a las necesidades de los usuarios.