

Práctica: Implementación de Base de Datos Distribuida para Gestión de Flota Vehicular

Asignatura: Bases de Datos Distribuidas

Catedrático: Dr. Eduardo Cornejo Velázquez

Equipo Integrante:

Yhojan Jesus Lozano Vazquez

Edwin Aaron Lopez Jimenez

Juan Pablo Ramos Hernández

Semestre: 6, Grupo: 2

Universidad Autónoma del Estado de Hidalgo

Índice

1. Introducción	3
2. Marco Teórico	3
2.1. Fragmentación Vertical	3
2.2. Procesos ETL (Extract, Transform, Load)	3
2.3. SELECT + INTO OUTFILE	4
2.4. LOAD DATA INFILE	4
2.5. SELECT con Tablas de Dos Bases de Datos	5
3. Implementación de Nodos Distribuidos	5
3.1. Esquema Conceptual Local de Cada Nodo	5
3.1.1. Nodo LCS1-Principal	5
3.1.2. Nodo LCS2-Mantenimiento	5
3.1.3. Nodo LCS3-Rutas	6
3.2. Script de Creación de Nodos	6
3.3. Scripts de Extracción de Datos	9
3.4. Script de Carga de Datos	13
3.5. Script de Consulta de Datos a Dos Tablas en Dos Nodos	15
4. Conclusiones	19

1 Introducción

Esta práctica tiene como objetivo implementar una base de datos distribuida para la gestión de flotas vehiculares, aplicando los conceptos de fragmentación vertical y procesos ETL. Se crearán tres nodos especializados que contendrán únicamente la información relevante para sus funciones específicas, optimizando así el rendimiento y la seguridad del sistema.

2 Marco Teórico

2.1 Fragmentación Vertical

Teoría: La fragmentación vertical es una técnica de diseño de bases de datos distribuidas que divide una tabla en subconjuntos de columnas, donde cada fragmento contiene atributos relacionados funcionalmente. Esto permite que diferentes aplicaciones o usuarios accedan solo a los datos relevantes para sus necesidades específicas.

Principales características:

- Cada fragmento debe contener la clave primaria para permitir reconstrucción
- Los fragmentos deben ser disjuntos en columnas (excepto la clave primaria)
- Optimiza el rendimiento al reducir transferencia de datos innecesarios
- Mejora la seguridad al limitar acceso a información sensible

Sintaxis:

```
1  -- Crear fragmento vertical
2  CREATE TABLE fragmento_vehiculo_financiero AS
3  SELECT id_auto, precio_compra, valor_actual, tasa_seguro
4  FROM vehiculo;
5
6  -- Fragmentación vertical con condiciones
7  CREATE TABLE fragmento_conductores_activos AS
8  SELECT id_conductor, nombre, licencia, telefono
9  FROM conductor
10 WHERE estado = 'Activo';
```

2.2 Procesos ETL (Extract, Transform, Load)

Teoría: Los procesos ETL son fundamentales en sistemas distribuidos para mover datos entre diferentes nodos. Consisten en tres fases:

- **Extracción (Extract):** Obtención de datos desde fuentes originales
- **Transformación (Transform):** Limpieza, validación y conversión de datos
- **Carga (Load):** Inserción de datos procesados en destinos

Aplicación en bases distribuidas:

```
1 -- Extracci n
2 SELECT * FROM tabla_fuente WHERE fecha > '2024-01-01';
3
4 -- Transformaci n
5 UPDATE temporal SET campo = UPPER(campo), fecha = DATE_FORMAT(fecha
6     , '%Y-%m-%d');
7
8 -- Carga
9 INSERT INTO tabla_destino SELECT * FROM temporal;
```

2.3 SELECT + INTO OUTFILE

Teoría: Comando de MySQL para exportar resultados de consultas a archivos externos en formato delimitado. Esencial para procesos de extracción en ETL.

Sintaxis completa:

```
1 SELECT columna1, columna2, columna3
2 INTO OUTFILE '/ruta/archivo.csv'
3 FIELDS TERMINATED BY ','
4 ENCLOSED BY '"'
5 ESCAPED BY '\\'
6 LINES TERMINATED BY '\n'
7 FROM tabla
8 WHERE condiciones;
```

Parámetros importantes:

- FIELDS TERMINATED BY: Delimitador de campos (ej: ',', '|', ';')
- ENCLOSED BY: Carácter para encerrar valores de texto
- LINES TERMINATED BY: Delimitador de líneas
- ESCAPED BY: Carácter de escape para valores especiales

2.4 LOAD DATA INFILE

Teoría: Comando para cargar datos desde archivos externos a tablas MySQL. Contraparte de INTO OUTFILE utilizado en la fase de carga de ETL.

Sintaxis completa:

```
1 LOAD DATA INFILE '/ruta/archivo.csv'
2 INTO TABLE tabla_destino
3 FIELDS TERMINATED BY ','
4 ENCLOSED BY '"'
5 ESCAPED BY '\\'
6 LINES TERMINATED BY '\n'
7 IGNORE 1 LINES
8 (columna1, columna2, columna3);
```

Parámetros importantes:

- **IGNORE n LINES:** Omite las primeras n líneas (útil para encabezados)
- **Lista de columnas:** Especifica el orden de columnas destino

2.5 SELECT con Tablas de Dos Bases de Datos

Teoría: MySQL permite consultar datos de múltiples bases de datos en una misma consulta, facilitando la integración en entornos distribuidos.

Sintaxis:

```
1 SELECT
2     t1.columna ,
3     t2.columna ,
4     t3.columna
5 FROM basedatos1.tabla1 t1
6 JOIN basedatos2.tabla2 t2 ON t1.id = t2.id
7 LEFT JOIN basedatos3.tabla3 t3 ON t1.id = t3.id
8 WHERE condiciones;
```

Consideraciones:

- Todas las bases de datos deben estar en el mismo servidor MySQL
- Se requieren permisos de acceso a todas las bases de datos involucradas
- El rendimiento depende de la distribución física de los datos

3 Implementación de Nodos Distribuidos

3.1 Esquema Conceptual Local de Cada Nodo

3.1.1. Nodo LCS1-Principal

Propósito: Gestión administrativa y financiera

Ubicación: Servidor Central

Esquema:

LCS1_Principal

flotilla (id_flotilla, nombre, descripcion, fecha_creacion)

vehiculo (id_auto, marca, modelo, anio, color, pasajeros, placa, estado)

documento (id_documento, id_auto, numero, nombre, inicio, final, monto)

Descripción: Este nodo contiene información administrativa y financiera de la flota vehicular. Incluye datos completos de los vehículos, documentación legal y configuración general de la flota.

3.1.2. Nodo LCS2-Mantenimiento

Propósito: Gestión técnica y de reparaciones

Ubicación: Taller de Mantenimiento

Esquema:

LCS2_Mantenimiento

```
vehiculo (id_auto, marca, modelo, anio, placa, kilometraje, ultimo_mantenimiento)
mantenimiento (id_mantenimiento, id_auto, fechainicio, fechafinal,
               diagnostico, descripcion, costo)
```

Descripción: Nodo especializado en información técnica y de mantenimiento. Contiene datos de kilometraje, historial de reparaciones y diagnósticos técnicos.

3.1.3. Nodo LCS3-Rutas

Propósito: Gestión operativa de rutas y conductores

Ubicación: Centro de Operaciones

Esquema:

LCS3_Rutas

```
vehiculo (id_auto, marca, modelo, placa, capacidad_combustible, consumo_promedio)
conductor (id_conductor, nombre, apellidopaterno, apellidomaterno, telefono, licen
ruta (id_ruta, id_auto, id_conductor, fecha, horainicio, origen, destino,
      horallegada, distancia_km)
transaccionCombustible (id_transaccion, id_auto, fecha, litros, costo,
                        odometro, estacion)
```

Descripción: Nodo operativo que gestiona rutas, conductores y consumo de combustible. Contiene información en tiempo real de las operaciones de transporte.

3.2 Script de Creación de Nodos

```
1  -- =====
2  -- NODO LCS1-PRINCIPAL: Gest i n Administrativa
3  -- =====
4
5  CREATE DATABASE IF NOT EXISTS LCS1_Principal;
6  USE LCS1_Principal;
7
8  -- Tabla de configuraci n de flotilla
9  CREATE TABLE flotilla (
10     id_flotilla INT PRIMARY KEY AUTO_INCREMENT,
11     nombre VARCHAR(255) NOT NULL,
12     descripcion TEXT,
13     fecha_creacion DATE DEFAULT CURRENT_DATE,
14     estado ENUM('Activa', 'Inactiva') DEFAULT 'Activa'
15 );
16
17 -- Tabla de veh culos con informaci n administrativa
18 CREATE TABLE vehiculo (
19     id_auto INT PRIMARY KEY,
20     marca VARCHAR(255) NOT NULL,
21     modelo VARCHAR(255) NOT NULL,
22     anio INT NOT NULL,
23     color VARCHAR(255) NOT NULL,
24     pasajeros INT NOT NULL,
```

```

25     placa VARCHAR(255) UNIQUE NOT NULL,
26     estado ENUM('Activo', 'Inactivo', 'Mantenimiento') DEFAULT '
        Activo',
27     fecha_adquisicion DATE,
28     valor_compra DECIMAL(12,2)
29 );
30
31 -- Tabla de documentaci n legal y financiera
32 CREATE TABLE documento (
33     id_documento INT PRIMARY KEY AUTO_INCREMENT,
34     id_auto INT NOT NULL,
35     numero VARCHAR(255) NOT NULL,
36     nombre VARCHAR(255) NOT NULL,
37     tipo ENUM('Tenencia', 'Seguro', 'Verificaci n', 'Permiso') NOT
        NULL,
38     inicio DATE NOT NULL,
39     final DATE NOT NULL,
40     monto DECIMAL(10,2) NOT NULL,
41     estado ENUM('Vigente', 'Vencido', 'En proceso') DEFAULT '
        Vigente',
42     FOREIGN KEY (id_auto) REFERENCES vehiculo(id_auto) ON DELETE
        CASCADE,
43     INDEX idx_auto (id_auto),
44     INDEX idx_vencimiento (final)
45 );
46
47 -- =====
48 -- NODO LCS2-MANTENIMIENTO: Gest i n T c n i c a
49 -- =====
50
51 CREATE DATABASE IF NOT EXISTS LCS2_Mantenimiento;
52 USE LCS2_Mantenimiento;
53
54 -- Tabla de veh culos con enfoque t c n i c o
55 CREATE TABLE vehiculo (
56     id_auto INT PRIMARY KEY,
57     marca VARCHAR(255) NOT NULL,
58     modelo VARCHAR(255) NOT NULL,
59     anio INT NOT NULL,
60     placa VARCHAR(255) UNIQUE NOT NULL,
61     kilometraje INT DEFAULT 0,
62     ultimo_mantenimiento DATE,
63     proximo_mantenimiento DATE,
64     estado_motor ENUM(' ptimo ', 'Regular', 'Cr t i c o') DEFAULT '
        ptimo ',
65     presion_neumaticos DECIMAL(4,2),
66     nivel_aceite ENUM('Alto', 'Medio', 'Bajo') DEFAULT 'Medio'
67 );
68
69 -- Tabla de historial de mantenimiento
70 CREATE TABLE mantenimiento (

```

```

71     id_mantenimiento INT PRIMARY KEY AUTO_INCREMENT,
72     id_auto INT NOT NULL,
73     fechainicio DATE NOT NULL,
74     fechafinal DATE NOT NULL,
75     tipo ENUM('Preventivo', 'Correctivo', 'Predictivo') NOT NULL,
76     diagnostico VARCHAR(255) NOT NULL,
77     descripcion TEXT NOT NULL,
78     costo DECIMAL(10,2) NOT NULL,
79     taller VARCHAR(255),
80     tecnico_responsable VARCHAR(255),
81     FOREIGN KEY (id_auto) REFERENCES vehiculo(id_auto) ON DELETE
      CASCADE,
82     INDEX idx_auto_fecha (id_auto, fechainicio),
83     INDEX idx_tipo (tipo)
84 );
85
86 -- =====
87 -- NODO LCS3-RUTAS: Gest i n Operativa
88 -- =====
89
90 CREATE DATABASE IF NOT EXISTS LCS3_Rutas;
91 USE LCS3_Rutas;
92
93 -- Tabla de veh culos con datos operativos
94 CREATE TABLE vehiculo (
95     id_auto INT PRIMARY KEY,
96     marca VARCHAR(255) NOT NULL,
97     modelo VARCHAR(255) NOT NULL,
98     placa VARCHAR(255) UNIQUE NOT NULL,
99     capacidad_combustible DECIMAL(8,2) NOT NULL,
100     consumo_promedio DECIMAL(6,2) NOT NULL,
101     odometro_actual INT DEFAULT 0,
102     disponible BOOLEAN DEFAULT TRUE,
103     ubicacion_actual VARCHAR(255)
104 );
105
106 -- Tabla de conductores
107 CREATE TABLE conductor (
108     id_conductor INT PRIMARY KEY,
109     nombre VARCHAR(255) NOT NULL,
110     apellidopaterno VARCHAR(255) NOT NULL,
111     apellidomaterno VARCHAR(255) NOT NULL,
112     telefono VARCHAR(255) NOT NULL,
113     licencia VARCHAR(255) UNIQUE NOT NULL,
114     tipo_licencia ENUM('A', 'B', 'C', 'D') NOT NULL,
115     vencimiento_licencia DATE NOT NULL,
116     activo BOOLEAN DEFAULT TRUE,
117     INDEX idx_licencia (licencia),
118     INDEX idx_activo (activo)
119 );
120

```



```

121 -- Tabla de rutas y trayectos
122 CREATE TABLE ruta (
123     id_ruta INT PRIMARY KEY AUTO_INCREMENT,
124     id_auto INT NOT NULL,
125     id_conductor INT NOT NULL,
126     fecha DATE NOT NULL,
127     horainicio TIME NOT NULL,
128     origen VARCHAR(255) NOT NULL,
129     destino VARCHAR(255) NOT NULL,
130     horallegada TIME,
131     distancia_km DECIMAL(8,2) NOT NULL,
132     estado ENUM('Programada', 'En curso', 'Completada', 'Cancelada'
133         ) DEFAULT 'Programada',
134     observaciones TEXT,
135     FOREIGN KEY (id_auto) REFERENCES vehiculo(id_auto),
136     FOREIGN KEY (id_conductor) REFERENCES conductor(id_conductor),
137     INDEX idx_fecha (fecha),
138     INDEX idx_auto_conductor (id_auto, id_conductor),
139     INDEX idx_estado (estado)
140 );
141
142 -- Tabla de control de combustible
143 CREATE TABLE transaccionCombustible (
144     id_transaccion INT PRIMARY KEY AUTO_INCREMENT,
145     id_auto INT NOT NULL,
146     fecha DATE NOT NULL,
147     litros DECIMAL(6,2) NOT NULL,
148     costo DECIMAL(8,2) NOT NULL,
149     odometro INT NOT NULL,
150     estacion VARCHAR(255) NOT NULL,
151     tipo_combustible ENUM('Magna', 'Premium', 'Di sel') NOT NULL,
152     rendimiento DECIMAL(6,2),
153     FOREIGN KEY (id_auto) REFERENCES vehiculo(id_auto),
154     INDEX idx_auto_fecha (id_auto, fecha),
155     INDEX idx_fecha (fecha)
156 );

```

Listing 1: Script de Creación de Nodos Distribuidos

3.3 Scripts de Extracción de Datos

```

1 -- =====
2 -- EXTRACCI N PARA LCS1-PRINCIPAL
3 -- =====
4
5 -- Extracci n de veh culos para nodo principal
6 SELECT
7     id_auto,
8     marca,
9     modelo,
10    anio,

```

```

11     color,
12     pasajeros,
13     placa,
14     'Activo' as estado,
15     CURDATE() as fecha_adquisicion,
16     ROUND(RAND() * 500000 + 100000, 2) as valor_compra
17 FROM FlotaVehicular.auto
18 INTO OUTFILE '/tmp/vehiculos_principal.csv'
19 FIELDS TERMINATED BY '|'
20 ENCLOSED BY '"'
21 ESCAPED BY '\\
22 LINES TERMINATED BY '\n';
23
24 -- Extracci n de documentos
25 SELECT
26     id_documento,
27     id_auto,
28     numero,
29     nombre,
30     CASE
31         WHEN nombre LIKE '%Tenencia%' THEN 'Tenencia'
32         WHEN nombre LIKE '%Seguro%' THEN 'Seguro'
33         WHEN nombre LIKE '%Verificacion%' THEN 'Verificaci n'
34         ELSE 'Permiso'
35     END as tipo,
36     inicio,
37     final,
38     monto,
39     CASE
40         WHEN final > CURDATE() THEN 'Vigente'
41         ELSE 'Vencido'
42     END as estado
43 FROM FlotaVehicular.documento
44 INTO OUTFILE '/tmp/documentos.csv'
45 FIELDS TERMINATED BY '|'
46 ENCLOSED BY '"'
47 ESCAPED BY '\\
48 LINES TERMINATED BY '\n';
49
50 -- =====
51 -- EXTRACCI N PARA LCS2-MANTENIMIENTO
52 -- =====
53
54 -- Extracci n de veh culos para mantenimiento
55 SELECT
56     id_auto,
57     marca,
58     modelo,
59     anio,
60     placa,
61     FLOOR(RAND() * 100000) as kilometraje,

```

```

62     DATE_SUB(CURDATE(), INTERVAL FLOOR(RAND() * 180) DAY) as
        ultimo_mantenimiento,
63     DATE_ADD(CURDATE(), INTERVAL FLOOR(RAND() * 90) DAY) as
        proximo_mantenimiento,
64     CASE FLOOR(RAND() * 3)
65         WHEN 0 THEN 'ptimo'
66         WHEN 1 THEN 'Regular'
67         ELSE 'Cr tico'
68     END as estado_motor,
69     ROUND(RAND() * 10 + 28, 2) as presion_neumaticos,
70     CASE FLOOR(RAND() * 3)
71         WHEN 0 THEN 'Alto'
72         WHEN 1 THEN 'Medio'
73         ELSE 'Bajo'
74     END as nivel_aceite
75 FROM FlotaVehicular.auto
76 INTO OUTFILE '/tmp/vehiculos_mantenimiento.csv'
77 FIELDS TERMINATED BY '|'
78 ENCLOSED BY '"'
79 ESCAPED BY '\\'
80 LINES TERMINATED BY '\n';
81
82 -- Extracci n de mantenimientos
83 SELECT
84     id_mantenimiento,
85     id_auto,
86     fechainicio,
87     fechafinal,
88     CASE FLOOR(RAND() * 3)
89         WHEN 0 THEN 'Preventivo'
90         WHEN 1 THEN 'Correctivo'
91         ELSE 'Predictivo'
92     END as tipo,
93     diagnostico,
94     description as descripcion,
95     ROUND(RAND() * 5000, 2) as costo,
96     CONCAT('Taller ', FLOOR(RAND() * 5 + 1)) as taller,
97     (SELECT CONCAT(nombre, ' ', apellidopaterno)
98      FROM FlotaVehicular.mecanico
99      ORDER BY RAND() LIMIT 1) as tecnico_responsable
100 FROM FlotaVehicular.mantenimiento
101 INTO OUTFILE '/tmp/mantenimientos.csv'
102 FIELDS TERMINATED BY '|'
103 ENCLOSED BY '"'
104 ESCAPED BY '\\'
105 LINES TERMINATED BY '\n';
106
107 -- =====
108 -- EXTRACCI N PARA LCS3-RUTAS
109 -- =====
110

```

```

111 -- Extracci n de veh culos para rutas
112 SELECT
113     id_auto,
114     marca,
115     modelo,
116     placa,
117     ROUND(RAND() * 80 + 20, 2) as capacidad_combustible,
118     ROUND(RAND() * 5 + 8, 2) as consumo_promedio,
119     FLOOR(RAND() * 100000) as odometro_actual,
120     TRUE as disponible,
121     CASE FLOOR(RAND() * 5)
122         WHEN 0 THEN 'Taller Central'
123         WHEN 1 THEN 'Base Norte'
124         WHEN 2 THEN 'Base Sur'
125         WHEN 3 THEN 'En ruta'
126         ELSE 'Estacionamiento'
127     END as ubicacion_actual
128 FROM FlotaVehicular.auto
129 INTO OUTFILE '/tmp/vehiculos_rutas.csv'
130 FIELDS TERMINATED BY '|'
131 ENCLOSED BY '"'
132 ESCAPED BY '\\'
133 LINES TERMINATED BY '\n';
134
135 -- Extracci n de conductores
136 SELECT
137     id_conductor,
138     nombre,
139     apellidopaterno,
140     apellidomaterno,
141     telefono,
142     licencia,
143     CASE FLOOR(RAND() * 4)
144         WHEN 0 THEN 'A'
145         WHEN 1 THEN 'B'
146         WHEN 2 THEN 'C'
147         ELSE 'D'
148     END as tipo_licencia,
149     DATE_ADD(CURDATE(), INTERVAL FLOOR(RAND() * 365) DAY) as
        vencimiento_licencia,
150     TRUE as activo
151 FROM FlotaVehicular.conductor
152 INTO OUTFILE '/tmp/conductores.csv'
153 FIELDS TERMINATED BY '|'
154 ENCLOSED BY '"'
155 ESCAPED BY '\\'
156 LINES TERMINATED BY '\n';
157
158 -- Extracci n de rutas
159 SELECT
160     id_ruta,

```

```

161     id_auto,
162     id_conductor,
163     fecha,
164     horainicio,
165     origen,
166     destino,
167     horallegada,
168     ROUND(RAND() * 500 + 50, 2) as distancia_km,
169     CASE FLOOR(RAND() * 4)
170         WHEN 0 THEN 'Programada'
171         WHEN 1 THEN 'En curso'
172         WHEN 2 THEN 'Completada'
173         ELSE 'Cancelada'
174     END as estado,
175     'Sin observaciones' as observaciones
176 FROM FlotaVehicular.ruta
177 INTO OUTFILE '/tmp/rutas.csv'
178 FIELDS TERMINATED BY '|'
179 ENCLOSED BY '"'
180 ESCAPED BY '\\'
181 LINES TERMINATED BY '\n';

```

Listing 2: Scripts de Extracción de Datos desde Base Central

3.4 Script de Carga de Datos

```

1  -- =====
2  -- CARGA EN LCS1-PRINCIPAL
3  -- =====
4  USE LCS1_Principal;
5
6  -- Cargar datos de veh culos
7  LOAD DATA INFILE '/tmp/vehiculos_principal.csv'
8  INTO TABLE vehiculo
9  FIELDS TERMINATED BY '|'
10 ENCLOSED BY '"'
11 ESCAPED BY '\\'
12 LINES TERMINATED BY '\n'
13 (id_auto, marca, modelo, anio, color, pasajeros, placa, estado,
14  fecha_adquisicion, valor_compra);
15
16 -- Cargar datos de documentos
17 LOAD DATA INFILE '/tmp/documentos.csv'
18 INTO TABLE documento
19 FIELDS TERMINATED BY '|'
20 ENCLOSED BY '"'
21 ESCAPED BY '\\'
22 LINES TERMINATED BY '\n'
23 (id_documento, id_auto, numero, nombre, tipo, inicio, final, monto,
24  estado);

```

```

25 -- Insertar configuraci n de flotilla
26 INSERT INTO flotilla (nombre, descripcion) VALUES
27 ('Flota Corporativa Principal', 'Flota vehicular para operaciones
    corporativas'),
28 ('Flota de Servicio T cnico', 'Veh culos dedicados a servicios
    t cnicos'),
29 ('Flota Ejecutiva', 'Veh culos para personal directivo');
30
31 -- =====
32 -- CARGA EN LCS2-MANTENIMIENTO
33 -- =====
34 USE LCS2_Mantenimiento;
35
36 -- Cargar datos de veh culos
37 LOAD DATA INFILE '/tmp/vehiculos_mantenimiento.csv'
38 INTO TABLE vehiculo
39 FIELDS TERMINATED BY '|'
40 ENCLOSED BY '"'
41 ESCAPED BY '\\'
42 LINES TERMINATED BY '\n'
43 (id_auto, marca, modelo, anio, placa, kilometraje,
    ultimo_mantenimiento,
44 proximo_mantenimiento, estado_motor, presion_neumaticos,
    nivel_aceite);
45
46 -- Cargar datos de mantenimiento
47 LOAD DATA INFILE '/tmp/mantenimientos.csv'
48 INTO TABLE mantenimiento
49 FIELDS TERMINATED BY '|'
50 ENCLOSED BY '"'
51 ESCAPED BY '\\'
52 LINES TERMINATED BY '\n'
53 (id_mantenimiento, id_auto, fechainicio, fechafinal, tipo,
    diagnostico,
54 descripcion, costo, taller, tecnico_responsable);
55
56 -- =====
57 -- CARGA EN LCS3-RUTAS
58 -- =====
59 USE LCS3_Rutas;
60
61 -- Cargar datos de veh culos
62 LOAD DATA INFILE '/tmp/vehiculos_rutas.csv'
63 INTO TABLE vehiculo
64 FIELDS TERMINATED BY '|'
65 ENCLOSED BY '"'
66 ESCAPED BY '\\'
67 LINES TERMINATED BY '\n'
68 (id_auto, marca, modelo, placa, capacidad_combustible,
    consumo_promedio,
69 odometro_actual, disponible, ubicacion_actual);

```

```

70
71 -- Cargar datos de conductores
72 LOAD DATA INFILE '/tmp/conductores.csv'
73 INTO TABLE conductor
74 FIELDS TERMINATED BY '|'
75 ENCLOSED BY '"'
76 ESCAPED BY '\\'
77 LINES TERMINATED BY '\n'
78 (id_conductor, nombre, apellidopaterno, apellidomaterno, telefono,
79  licencia,
80  tipo_licencia, vencimiento_licencia, activo);
81
82 -- Cargar datos de rutas
83 LOAD DATA INFILE '/tmp/rutas.csv'
84 INTO TABLE ruta
85 FIELDS TERMINATED BY '|'
86 ENCLOSED BY '"'
87 ESCAPED BY '\\'
88 LINES TERMINATED BY '\n'
89 (id_ruta, id_auto, id_conductor, fecha, horainicio, origen, destino
90  ,
91  horallegada, distancia_km, estado, observaciones);
92
93 -- Insertar transacciones de combustible de ejemplo
94 INSERT INTO transaccionCombustible
95 (id_auto, fecha, litros, costo, odometro, estacion,
96  tipo_combustible, rendimiento)
97 VALUES
98 (101, '2025-09-01', 40.5, 850.50, 15000, 'Estaci n Central', '
99  Premium', 12.5),
100 (102, '2025-09-02', 35.2, 740.20, 12000, 'Estaci n Norte', 'Magna'
101  , 11.8),
102 (103, '2025-09-03', 60.0, 1260.00, 8000, 'Estaci n Sur', 'Di sel'
103  , 9.5),
104 (104, '2025-09-04', 45.8, 961.80, 25000, 'Estaci n Este', 'Premium
105  ', 13.2),
106 (105, '2025-09-05', 38.7, 812.70, 18000, 'Estaci n Oeste', 'Magna'
107  , 12.1),
108 (101, '2025-09-08', 42.3, 888.30, 15500, 'Estaci n Central', '
109  Premium', 12.8),
110 (102, '2025-09-09', 36.8, 772.80, 12500, 'Estaci n Norte', 'Magna'
111  , 11.9);

```

Listing 3: Script de Carga de Datos en Nodos Distribuidos

3.5 Script de Consulta de Datos a Dos Tablas en Dos Nodos

```

1 -- =====
2 -- CONSULTA 1: Informaci n financiera y operativa combinada
3 -- Combina datos de LCS1-Principal y LCS3-Rutas
4 -- =====

```

```

5
6 SELECT
7     -- Datos del veh culo desde LCS1-Principal
8     v1.marca,
9     v1.modelo,
10    v1.placa,
11    v1.estado as estado_administrativo,
12    v1.valor_compra,
13
14    -- Datos de rutas desde LCS3-Rutas
15    COUNT(r.id_ruta) as total_rutas,
16    SUM(r.distancia_km) as kilometraje_total,
17    AVG(r.distancia_km) as distancia_promedio,
18
19    -- Datos de documentos desde LCS1-Principal
20    COUNT(d.id_documento) as documentos_activos,
21    SUM(CASE WHEN d.estado = 'Vencido' THEN 1 ELSE 0 END) as
22        documentos_vencidos,
23
24    -- Datos de combustible desde LCS3-Rutas
25    COUNT(tc.id_transaccion) as recargas_combustible,
26    SUM(tc.costo) as total_gasto_combustible,
27    AVG(tc.rendimiento) as rendimiento_promedio
28 FROM LCS1_Principal.vehiculo v1
29
30 -- Join con rutas del ltimo mes (LCS3-Rutas)
31 LEFT JOIN LCS3_Rutas.ruta r ON v1.id_auto = r.id_auto
32     AND r.fecha >= DATE_SUB(CURDATE(), INTERVAL 30 DAY)
33     AND r.estado = 'Completada'
34
35 -- Join con documentos (LCS1-Principal)
36 LEFT JOIN LCS1_Principal.documento d ON v1.id_auto = d.id_auto
37     AND d.final >= CURDATE()
38
39 -- Join con transacciones de combustible del ltimo mes (LCS3-
40     Rutas)
41 LEFT JOIN LCS3_Rutas.transaccionCombustible tc ON v1.id_auto = tc.
42     id_auto
43     AND tc.fecha >= DATE_SUB(CURDATE(), INTERVAL 30 DAY)
44
45 WHERE v1.estado = 'Activo'
46 GROUP BY v1.id_auto, v1.marca, v1.modelo, v1.placa, v1.estado, v1.
47     valor_compra
48 ORDER BY total_gasto_combustible DESC;
49
50 -- =====
51 -- CONSULTA 2: An lisis de mantenimiento y operaciones
52 -- Combina datos de LCS2-Mantenimiento y LCS3-Rutas
53 -- =====

```



```

52 SELECT
53     v2.marca,
54     v2.modelo,
55     v2.placa,
56     v2.kilometraje,
57     v2.ultimo_mantenimiento,
58     v2.estado_motor,
59     m.diagnostico,
60     m.costo as costo_mantenimiento,
61     COUNT(r.id_ruta) as total_rutas_mes,
62     SUM(r.distancia_km) as kilometraje_mes,
63     AVG(tc.rendimiento) as rendimiento_promedio,
64     SUM(tc.costo) as gasto_combustible_mes,
65     CASE
66         WHEN v2.estado_motor = 'Cr tico' THEN 'Mantenimiento
           Urgente'
67         WHEN v2.kilometraje > 80000 THEN 'Mantenimiento Programado'
68         ELSE 'En condiciones ptimas '
69     END as recomendacion
70
71 FROM LCS2_Mantenimiento.vehiculo v2
72
73 -- Join con mantenimientos recientes
74 LEFT JOIN LCS2_Mantenimiento.mantenimiento m ON v2.id_auto = m.
       id_auto
75         AND m.fechafinal >= DATE_SUB(CURDATE(), INTERVAL 60 DAY)
76
77 -- Join con rutas del ltimo mes
78 LEFT JOIN LCS3_Rutas.ruta r ON v2.id_auto = r.id_auto
79         AND r.fecha >= DATE_SUB(CURDATE(), INTERVAL 30 DAY)
80
81 -- Join con transacciones de combustible del ltimo mes
82 LEFT JOIN LCS3_Rutas.transaccionCombustible tc ON v2.id_auto = tc.
       id_auto
83         AND tc.fecha >= DATE_SUB(CURDATE(), INTERVAL 30 DAY)
84
85 GROUP BY v2.id_auto, v2.marca, v2.modelo, v2.placa, v2.kilometraje,
86         v2.ultimo_mantenimiento, v2.estado_motor, m.diagnostico, m
           .costo
87 ORDER BY v2.estado_motor, kilometraje_mes DESC;
88
89 -- =====
90 -- CONSULTA 3: Reporte ejecutivo integrado
91 -- Combina datos de los tres nodos
92 -- =====
93
94 SELECT
95     -- Datos b sicos del veh culo
96     v1.marca,
97     v1.modelo,
98     v1.placa,

```

```

99     v1.estado as estado_administrativo,
100
101     -- Datos de mantenimiento (LCS2)
102     v2.estado_motor,
103     v2.kilometraje,
104     v2.ultimo_mantenimiento,
105
106     -- Datos operativos (LCS3)
107     COUNT(DISTINCT r.id_ruta) as viajes_realizados,
108     SUM(r.distancia_km) as kilometraje_operativo,
109
110     -- Datos financieros (LCS1)
111     SUM(d.monto) as costo_documentos,
112     COUNT(d.id_documento) as total_documentos,
113
114     -- Datos de combustible (LCS3)
115     SUM(tc.costo) as costo_combustible,
116     AVG(tc.rendimiento) as eficiencia_combustible,
117
118     -- C lculo de costos totales
119     (SUM(COALESCE(d.monto, 0)) + SUM(COALESCE(tc.costo, 0))) as
        costo_total
120
121 FROM LCS1_Principal.vehiculo v1
122
123 -- Join con datos de mantenimiento
124 LEFT JOIN LCS2_Mantenimiento.vehiculo v2 ON v1.id_auto = v2.id_auto
125
126 -- Join con documentos
127 LEFT JOIN LCS1_Principal.documento d ON v1.id_auto = d.id_auto
128     AND d.final >= DATE_SUB(CURDATE(), INTERVAL 365 DAY)
129
130 -- Join con rutas del ltimo trimestre
131 LEFT JOIN LCS3_Rutas.ruta r ON v1.id_auto = r.id_auto
132     AND r.fecha >= DATE_SUB(CURDATE(), INTERVAL 90 DAY)
133     AND r.estado = 'Completada'
134
135 -- Join con combustible del ltimo trimestre
136 LEFT JOIN LCS3_Rutas.transaccionCombustible tc ON v1.id_auto = tc.
    id_auto
137     AND tc.fecha >= DATE_SUB(CURDATE(), INTERVAL 90 DAY)
138
139 WHERE v1.estado = 'Activo'
140 GROUP BY v1.id_auto, v1.marca, v1.modelo, v1.placa, v1.estado,
    v2.estado_motor, v2.kilometraje, v2.ultimo_mantenimiento
141 HAVING viajes_realizados > 0 OR total_documentos > 0
142 ORDER BY costo_total DESC, kilometraje_operativo DESC;
143

```

Listing 4: Script de Consultas entre Múltiples Nodos

4 Conclusiones

La implementación de esta base de datos distribuida demuestra la aplicación práctica de los conceptos de bases de datos distribuidas mediante:

- **Fragmentación vertical** aplicada a los diferentes nodos según su función específica
- **Procesos ETL** completos para la extracción, transformación y carga de datos entre nodos
- Uso eficiente de **SELECT INTO OUTFILE** y **LOAD DATA INFILE** para transferencia de datos
- **Consultas distribuidas** complejas que acceden a múltiples nodos simultáneamente

La arquitectura resultante ofrece ventajas significativas:

- **Optimización del rendimiento** al distribuir la carga de trabajo
- **Mejora de la seguridad** mediante el control de acceso a datos sensibles
- **Facilidad de mantenimiento** al separar responsabilidades por nodos
- **Escalabilidad** para futuras expansiones del sistema

Esta implementación sirve como base sólida para sistemas de gestión de flotas vehiculares en entornos empresariales distribuidos.