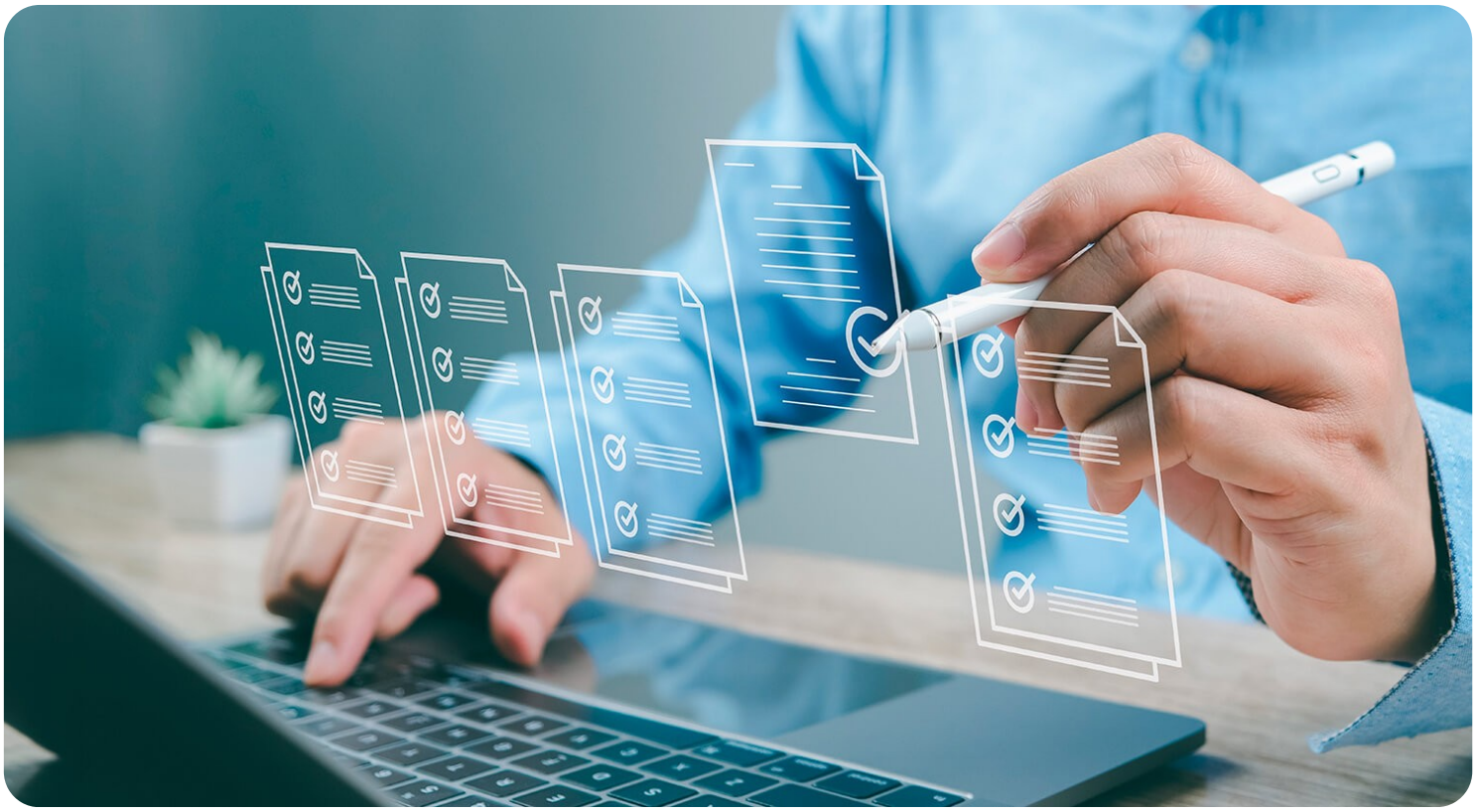


Semana 6

Modelamiento de Bases de Datos (PRY2204)

Formato de respuesta

Nombre estudiante:	Victor Vasquez Agurto
Asignatura: PRY2204	**Carrera:** Desarrollo de Aplicaciones
Profesor: Josué Oteiza	**Fecha:** 2024-09-16

Descripción de la actividad

En esta sexta semana, realizarás la actividad sumativa de forma individual, llamada "Generando claves primarias y foráneas en el Diseño físico", donde a través de un caso planteado, trabajarás directamente en el desarrollo de un modelo de base de datos, desarrollándolo paso a paso y aplicando normalización e integridad referencial para mostrar, finalmente, un Diseño físico del proyecto.

Instrucciones específicas

Para la ejecución de esta actividad, lee por última vez el caso de la cadena de servicios automotrices:

“El proyecto para la cadena de servicios automotrices en Santiago se embarca en una misión crucial: transformar y modernizar sus procesos operativos frente a los crecientes desafíos derivados del aumento en la clientela y la demanda de servicios. La gestión manual de información crucial, como los datos personales de los clientes, los presupuestos de mantenimiento o reparación, y el registro de servicios en papel, ha comenzado a afectar negativamente la calidad del servicio, causando una sobrecarga laboral para el personal, la pérdida de clientes y un general desorden administrativo.

En la realización del proyecto, te enfocaste en el modelado y la normalización de datos. Este proceso implicó identificar las entidades clave, como clientes y vehículos, y definir las relaciones esenciales, por ejemplo, cómo se vinculan las órdenes de atención con clientes y vehículos. La asignación de claves primarias y foráneas fue crucial para asegurar la unicidad de los registros y facilitar las relaciones entre las entidades. El objetivo era minimizar la redundancia de datos y maximizar la eficiencia operativa mediante la aplicación de las reglas de normalización (1FN, 2FN y 3FN).

Luego, con el modelo conceptual como base, avanzaste hacia la representación detallada del Modelo Entidad-Relación (MER), refinando entidades y atributos y asegurando la correcta asignación de claves. Este paso fue fundamental para transformar el MER en un modelo relacional detallado, preparando así la estructura para su futura implementación en el sistema de gestión de bases de datos.

Ahora, tendrás que centrar tu enfoque en generar claves primarias y foráneas dentro del diseño físico. Partiendo del MER y el modelo relacional normalizado desarrollados anteriormente, tu tarea será desarrollar el modelo de base de datos de manera más concreta. Este paso implica una cuidadosa consideración de la normalización e integridad referencial para culminar con un diseño físico del proyecto que sea tanto funcional como optimizado".

Para poder realizar todo el proceso, te brindaremos el paso a paso desde el punto de inicio:

Paso 1: Identificación de entidades y atributos

Comienza analizando el escenario operativo de la cadena de servicios automotrices para identificar entidades clave tales como Clientes, Vehículos, Órdenes de Servicio y Repuestos. Determina los atributos necesarios para cada entidad.

Paso 2: Definición de claves

Asigna una clave primaria única a cada entidad para asegurar registros únicos y define claves foráneas para establecer relaciones entre las entidades.

Paso 3: Establecimiento de relaciones

Clarifica cómo se relacionan las entidades identificadas entre sí, como la conexión entre Órdenes de Atención, Clientes y Vehículos.

Paso 4: Normalización

Aplica las reglas de normalización (1FN, 2FN, 3FN) para organizar el modelo de datos, reduciendo redundancias y asegurando la integridad de los datos.

Paso 5: Refinamiento de entidades y atributos

Revisa y ajusta las entidades y sus atributos para mayor claridad y precisión, basándote en el trabajo inicial.

Paso 6: Revisión y asignación de claves

Verifica y ajusta las claves primarias y foráneas, poniendo especial atención en la integridad referencial.

Paso 7: Transformación del MER a Modelo Relacional

Convierte el MER refinado en un modelo relacional detallado, preparando la estructura para su futura implementación.

Paso 8: Desarrollo del Modelo Identidad Relación Extendido (MERE)

Utiliza Oracle SQL Developer Data Modeler para crear un MERE que represente los campos opcionales y obligatorios, asegurando la normalización de las entidades.

- Puedes descargar Oracle SQL Data Modeler en el siguiente enlace:

https://www.oracle.com/database/sqldeveloper/technologies/sql-data-modeler/download/

Paso 9: Diseño del Modelo Relacional Normalizado (MR)

Construye el MR asegurando la consistencia con el MERE. Este modelo debe incluir todas las tablas, columnas, relaciones, claves primarias, llaves foráneas y destacar la obligatoriedad de las columnas.

Paso 10: Implementación de claves primarias y foráneas en el Diseño Físico

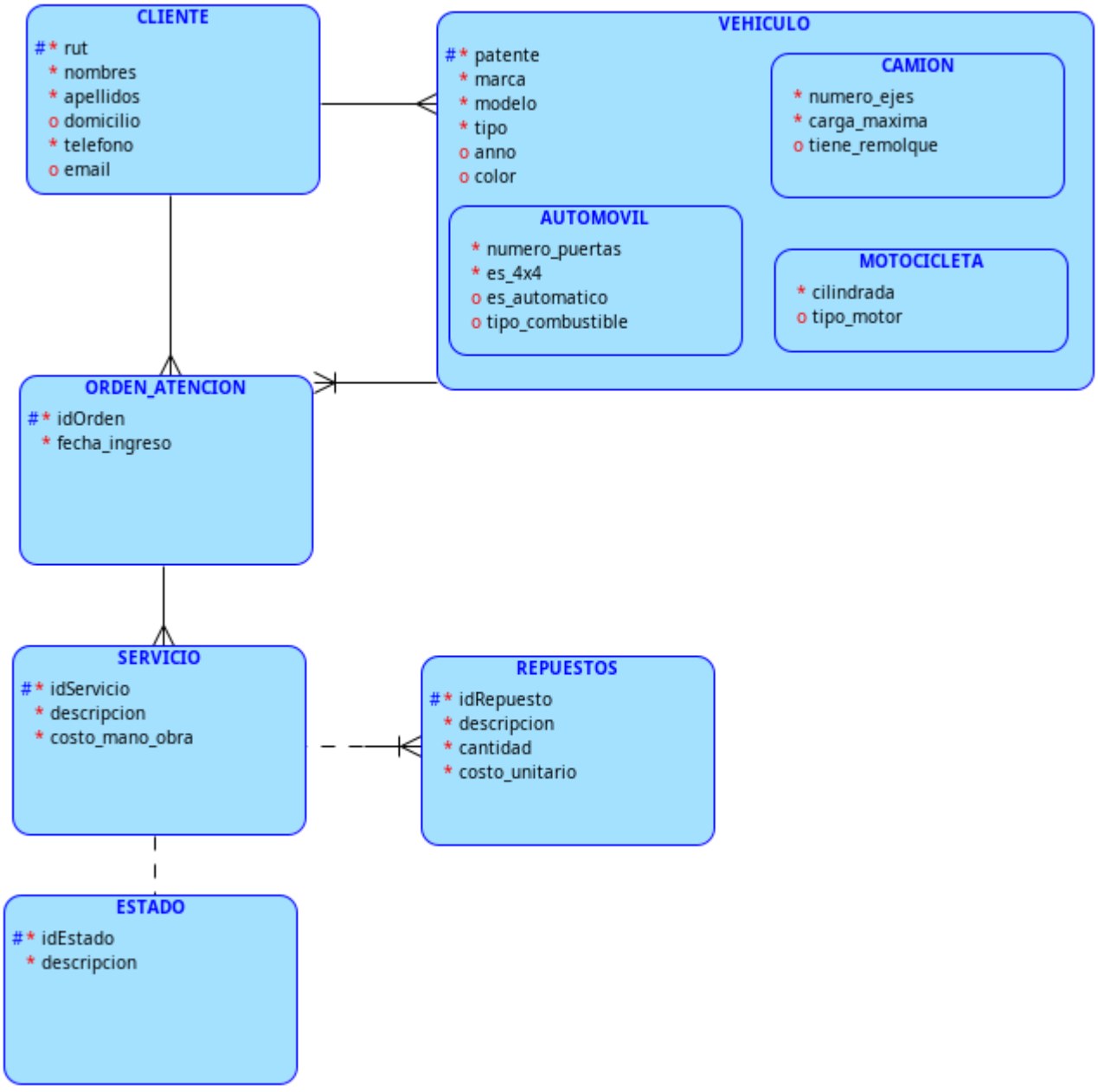
Especifica detalladamente las claves primarias y foráneas para cada tabla en el diseño físico, garantizando la integridad referencial y la eficiencia operativa del sistema.

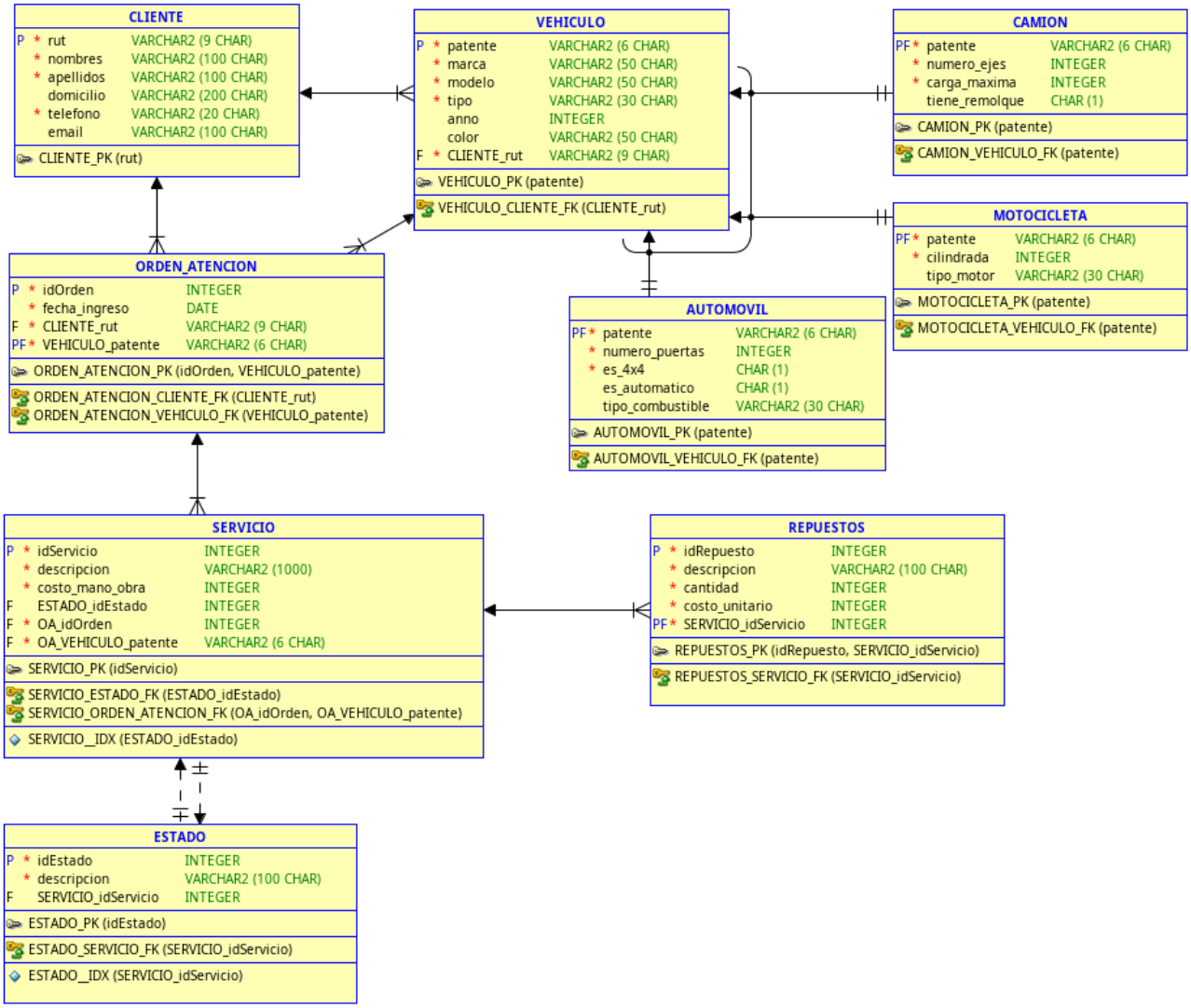
Paso 11: Optimización y documentación

Optimiza el diseño físico para mejorar el rendimiento del sistema y documenta el proceso de implementación de las claves. Valida el diseño físico mediante la generación de SQL scripts desde Oracle SQL Data Modeler.

Adjunta las imágenes solicitadas en los siguientes espacios:

1. **Modelo Identidad Relación Extendido (MERE)**




CLIENTE
P * rut VARCHAR2 (9 CHAR)
* nombres VARCHAR2 (100 CHAR)
* apellidos VARCHAR2 (100 CHAR)
domicilio VARCHAR2 (200 CHAR)
* telefono VARCHAR2 (20 CHAR)
email VARCHAR2 (100 CHAR)
CLIENTE_PK (rut)
VEHICULO
P * patente VARCHAR2 (6 CHAR)
* marca VARCHAR2 (50 CHAR)
* modelo VARCHAR2 (50 CHAR)
* tipo VARCHAR2 (30 CHAR)
anno INTEGER
color VARCHAR2 (50 CHAR)
F * CLIENTE_rut VARCHAR2 (9 CHAR)
VEHICULO_PK (patente)
VEHICULO_CLIENTE_FK (CLIENTE_rut)
CAMION
PF * patente VARCHAR2 (6 CHAR)
* numero_ejes INTEGER
* carga_maxima INTEGER
tiene_remolque CHAR (1)
CAMION_PK (patente)
CAMION_VEHICULO_FK (patente)
MOTOCICLETA
PF * patente VARCHAR2 (6 CHAR)
* cilindrada INTEGER
tipo_motor VARCHAR2 (30 CHAR)
MOTOCICLETA_PK (patente)
MOTOCICLETA_VEHICULO_FK (patente)
AUTOMOVIL
PF * patente VARCHAR2 (6 CHAR)
* numero_puertas INTEGER
* es_4x4 CHAR (1)
es_automatico CHAR (1)
tipo_combustible VARCHAR2 (30 CHAR)
AUTOMOVIL_PK (patente)
AUTOMOVIL_VEHICULO_FK (patente)
ORDEN_ATENCION
P * idOrden INTEGER
* fecha_ingreso DATE
F * CLIENTE_rut VARCHAR2 (9 CHAR)
PF * VEHICULO_patente VARCHAR2 (6 CHAR)
ORDEN_ATENCION_PK (idOrden, VEHICULO_patente)
ORDEN_ATENCION_CLIENTE_FK (CLIENTE_rut)
ORDEN_ATENCION_VEHICULO_FK (VEHICULO_patente)
SERVICIO
P * idServicio INTEGER
* descripcion VARCHAR2 (1000)
* costo_mano_obra INTEGER
F ESTADO_idEstado INTEGER
F * OA_idOrden INTEGER
F * OA_VEHICULO_patente VARCHAR2 (6 CHAR)
SERVICIO_PK (idServicio)
SERVICIO_ESTADO_FK (ESTADO_idEstado)
SERVICIO_ORDEN_ATENCION_FK (OA_idOrden, OA_VEHICULO_patente)
SERVICIO__IDX (ESTADO_idEstado)
REPUESTOS
P * idRepuesto INTEGER
* descripcion VARCHAR2 (100 CHAR)
* cantidad INTEGER
* costo_unitario INTEGER
PF * SERVICIO_idServicio INTEGER
REPUESTOS_PK (idRepuesto, SERVICIO_idServicio)
REPUESTOS_SERVICIO_FK (SERVICIO_idServicio)
ESTADO
P * idEstado INTEGER
* descripcion VARCHAR2 (100 CHAR)
F SERVICIO_idServicio INTEGER
ESTADO_PK (idEstado)
ESTADO_SERVICIO_FK (SERVICIO_idServicio)
ESTADO__IDX (SERVICIO_idServicio)

2. Modelo Relacional (MR)

```
CREATE TABLE automovil (
    patente          VARCHAR2(6 CHAR) NOT NULL,
    numero_puertas   INTEGER NOT NULL,
    es_4x4           NUMBER NOT NULL,
    es_automatico    NUMBER,
    tipo_combustible VARCHAR2(30 CHAR)
);

ALTER TABLE automovil ADD CONSTRAINT automovil_pk PRIMARY KEY ( patente );

CREATE TABLE camion (
    patente        VARCHAR2(6 CHAR) NOT NULL,
    numero_ejes    INTEGER NOT NULL,
    carga_maxima   INTEGER NOT NULL,
    tiene_remolque NUMBER
);

ALTER TABLE camion ADD CONSTRAINT camion_pk PRIMARY KEY ( patente );

CREATE TABLE cliente (
    rut       VARCHAR2(9 CHAR) NOT NULL,
    nombres   VARCHAR2(100 CHAR) NOT NULL,
    apellidos VARCHAR2(100 CHAR) NOT NULL,
    domicilio VARCHAR2(200 CHAR),
    telefono  VARCHAR2(20 CHAR) NOT NULL,
    email     VARCHAR2(100 CHAR)
);

ALTER TABLE cliente ADD CONSTRAINT cliente_pk PRIMARY KEY ( rut );

CREATE TABLE estado (
    idestado            INTEGER NOT NULL,
    descripcion         VARCHAR2(100 CHAR) NOT NULL,
    servicio_idservicio INTEGER
);

CREATE UNIQUE INDEX estado__idx ON
    estado (
        servicio_idservicio
    ASC );

ALTER TABLE estado ADD CONSTRAINT estado_pk PRIMARY KEY ( idestado );

CREATE TABLE motocicleta (
    patente    VARCHAR2(6 CHAR) NOT NULL,
    cilindrada INTEGER NOT NULL,
    tipo_motor VARCHAR2(30 CHAR)
);

ALTER TABLE motocicleta ADD CONSTRAINT motocicleta_pk PRIMARY KEY ( patente );

CREATE TABLE orden_atencion (
    idorden          INTEGER NOT NULL,
    fecha_ingreso    DATE NOT NULL,
    cliente_rut      VARCHAR2(9 CHAR) NOT NULL,
    vehiculo_patente VARCHAR2(6 CHAR) NOT NULL
);

ALTER TABLE orden_atencion ADD CONSTRAINT orden_atencion_pk PRIMARY KEY ( idorden,
                                                                          vehiculo_patente );
```

```
CREATE TABLE repuestos (
    idrepuesto          INTEGER NOT NULL,
    descripcion         VARCHAR2(100 CHAR) NOT NULL,
    cantidad            INTEGER NOT NULL,
    costo_unitario      INTEGER NOT NULL,
    servicio_idservicio INTEGER NOT NULL
);

ALTER TABLE repuestos ADD CONSTRAINT repuestos_pk PRIMARY KEY ( idrepuesto,
                                                               servicio_idservicio );

CREATE TABLE servicio (
    idservicio          INTEGER NOT NULL,
    descripcion         VARCHAR2(1000) NOT NULL,
    costo_mano_obra     INTEGER NOT NULL,
    estado_idestado     INTEGER,
    oa_idorden          INTEGER NOT NULL,
    oa_vehiculo_patente VARCHAR2(6 CHAR) NOT NULL
);

CREATE UNIQUE INDEX servicio__idx ON
    servicio (
        estado_idestado
    ASC );

ALTER TABLE servicio ADD CONSTRAINT servicio_pk PRIMARY KEY ( idservicio );

CREATE TABLE vehiculo (
    patente     VARCHAR2(6 CHAR) NOT NULL,
    marca       VARCHAR2(50 CHAR) NOT NULL,
    modelo      VARCHAR2(50 CHAR) NOT NULL,
    tipo        VARCHAR2(30 CHAR) NOT NULL,
    anno        INTEGER,
    color       VARCHAR2(50 CHAR),
    cliente_rut VARCHAR2(9 CHAR) NOT NULL
);

ALTER TABLE vehiculo ADD CONSTRAINT vehiculo_pk PRIMARY KEY ( patente );

ALTER TABLE automovil
    ADD CONSTRAINT automovil_vehiculo_fk FOREIGN KEY ( patente )
        REFERENCES vehiculo ( patente );

ALTER TABLE camion
    ADD CONSTRAINT camion_vehiculo_fk FOREIGN KEY ( patente )
        REFERENCES vehiculo ( patente );

ALTER TABLE estado
    ADD CONSTRAINT estado_servicio_fk FOREIGN KEY ( servicio_idservicio )
        REFERENCES servicio ( idservicio );

ALTER TABLE motocicleta
    ADD CONSTRAINT motocicleta_vehiculo_fk FOREIGN KEY ( patente )
        REFERENCES vehiculo ( patente );

ALTER TABLE orden_atencion
    ADD CONSTRAINT orden_atencion_cliente_fk FOREIGN KEY ( cliente_rut )
        REFERENCES cliente ( rut );

ALTER TABLE orden_atencion
    ADD CONSTRAINT orden_atencion_vehiculo_fk FOREIGN KEY ( vehiculo_patente )
        REFERENCES vehiculo ( patente );
```

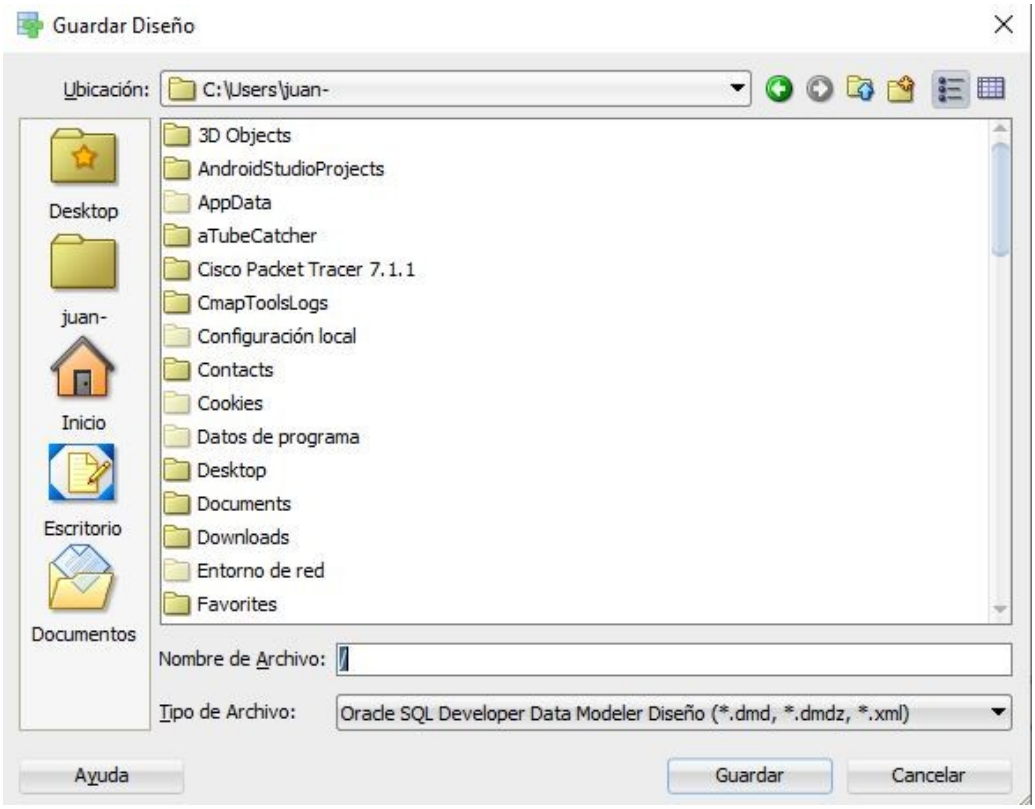
```
ALTER TABLE repuestos
    ADD CONSTRAINT repuestos_servicio_fk FOREIGN KEY ( servicio_idservicio )
        REFERENCES servicio ( idservicio );

ALTER TABLE servicio
    ADD CONSTRAINT servicio_estado_fk FOREIGN KEY ( estado_idestado )
        REFERENCES estado ( idestado );

ALTER TABLE servicio
    ADD CONSTRAINT servicio_orden_atencion_fk FOREIGN KEY ( oa_idorden,
                                                            oa_vehiculo_patente )
        REFERENCES orden_atencion ( idorden,
                                    vehiculo_patente );

ALTER TABLE vehiculo
    ADD CONSTRAINT vehiculo_cliente_fk FOREIGN KEY ( cliente_rut )
        REFERENCES cliente ( rut );

CREATE OR REPLACE TRIGGER arc_arco_tipo_vehi_motocicleta BEFORE
    INSERT OR UPDATE OF patente ON motocicleta
    FOR EACH ROW
DECLARE
    d VARCHAR2(30 CHAR);
BEGIN
    SELECT
        a.tipo
    INTO d
    FROM
        vehiculo a
    WHERE
        a.patente = :new.patente;

    IF ( d IS NULL OR d <> 'MOTOCICLETA' ) THEN
        raise_application_error(
                                -20223,
                                'FK MOTOCICLETA_VEHICULO_FK in Table MOTOCICLETA violates Arc
constraint on Table VEHICULO - discriminator column tipo doesn''t have value ''MOTOCICLETA'''
        );
    END IF;

EXCEPTION
    WHEN no_data_found THEN
        NULL;
    WHEN OTHERS THEN
        RAISE;
END;
/
```

```
CREATE OR REPLACE TRIGGER arc_arco_tipo_vehicu_automovil BEFORE
    INSERT OR UPDATE OF patente ON automovil
    FOR EACH ROW
DECLARE
    d VARCHAR2(30 CHAR);
BEGIN
    SELECT
        a.tipo
    INTO d
    FROM
        vehiculo a
    WHERE
        a.patente = :new.patente;

    IF ( d IS NULL OR d <> 'AUTOMOVIL' ) THEN
        raise_application_error(
                                -20223,
                                'FK AUTOMOVIL_VEHICULO_FK in Table AUTOMOVIL violates Arc constraint
on Table VEHICULO - discriminator column tipo doesn''t have value ''AUTOMOVIL'''
        );
    END IF;

EXCEPTION
    WHEN no_data_found THEN
        NULL;
    WHEN OTHERS THEN
        RAISE;
END;
/

CREATE OR REPLACE TRIGGER arc_arco_tipo_vehiculo_camion BEFORE
    INSERT OR UPDATE OF patente ON camion
    FOR EACH ROW
DECLARE
    d VARCHAR2(30 CHAR);
BEGIN
    SELECT
        a.tipo
    INTO d
    FROM
        vehiculo a
    WHERE
        a.patente = :new.patente;

    IF ( d IS NULL OR d <> 'CAMION' ) THEN
        raise_application_error(
                                -20223,
                                'FK CAMION_VEHICULO_FK in Table CAMION violates Arc constraint on
Table VEHICULO - discriminator column tipo doesn''t have value ''CAMION'''
        );
    END IF;

EXCEPTION
    WHEN no_data_found THEN
        NULL;
    WHEN OTHERS THEN
        RAISE;
END;
/
```

Paso 12: Tendrás que descargar el resultado. Para ello, tendrás que hacer clic en la opción Guardar como... del menú Archivo, esto despliega el submenú que se ilustra en la siguiente figura:

Figura 1

Cómo guardar un archivo en SQL



Nota. Ejemplo de guardado de archivo SQL. Oracle. (s.f.). *Oracle Data Modeler* [Software]. Oracle. https://www.oracle.com/cl/database/sqldeveloper/technologies/sql-data-modeler/

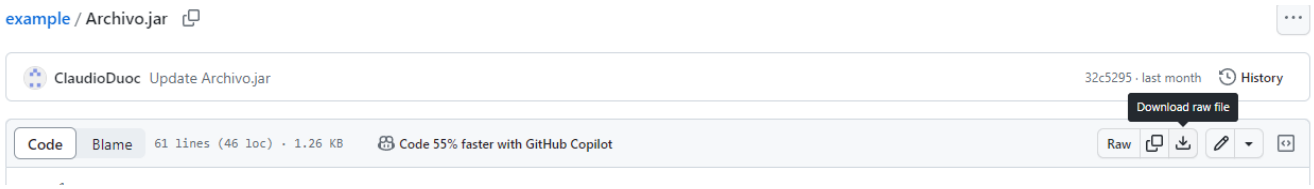
Paso 13: El archivo descargado desde SQL deberás subirlo al repositorio GitHub. Si no has creado tu cuenta aún, puedes hacerlo a través del siguiente enlace:

https://github.com/

Una vez subido el archivo a GitHub, deberás descargar el archivo comprimido .java desde tu repositorio, tal como se muestra en la imagen:

Figura 2

Archivo .raw en GitHub

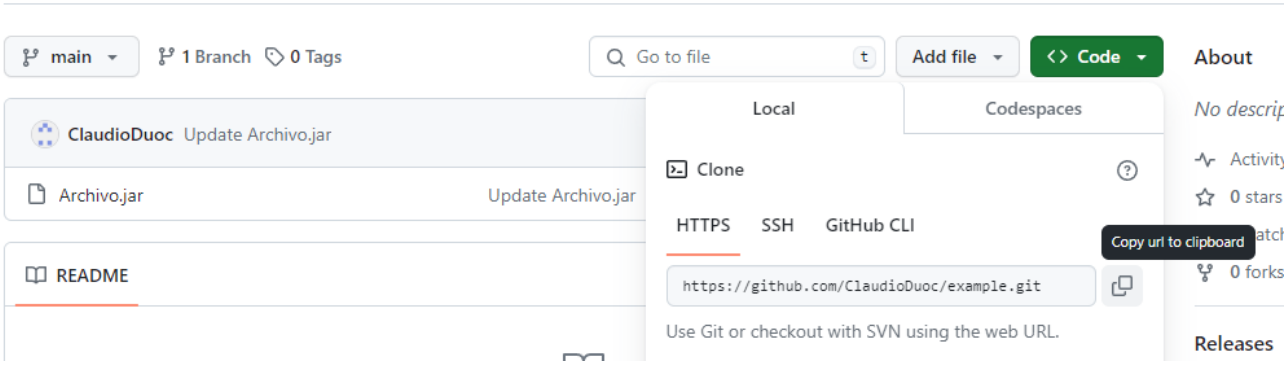


Nota. Descarga de archivo desde repositorio GitHub. GitHub (s.f.). *GitHub.* https://github.com/

Posteriormente, desde el repositorio, deberás generar un enlace de tu proyecto:

Figura 3

Enlace de proyecto GitHub



Nota. Ejemplo de dónde se extrae un enlace en GitHub. GitHub (s.f.). *GitHub.* https://github.com/

Deja en este apartado el enlace de tu repositorio GitHub:

https://github.com/vvasquez-duoc/PRY2204_Exp2_S6_Victor_Vasquez

Paso 14: una vez adjuntas tu respuestas y enlace, no olvides comprimir este documento y el archivo .raw y SQL en un archivo .rar, el cual deberás subir al AVA.