**PROYECTO FINAL**
**CASO DE ESTUDIO: CAR-CONTROL ONLINE**

### 1.1. CASO DE ESTUDIO.

Car Control Online es una empresa de la iniciativa privada que esta proponiendo un nuevo modelo al gobierno del país para modernizar y mejorar el control vehicular del país. La base de datos que propone para realizar este proyecto define las siguientes reglas de negocio.

El proceso inicia con la definición del catálogo de entidades del país (clave y nombre). La empresa propone el registro global de un catálogo de placas del automóvil. Se registra el número de placa (hasta 10 caracteres), la entidad a la que pertenece, fecha de asignación. Una placa puede ser asignada a un solo auto. Si las placas se dan de baja por parte del dueño del vehículo la placa se marca como inactiva y esta ya no se puede reutilizar. Notar que pueden existir en el catálogo, placas sin ser aun asignadas a un vehículo.
¿Como?
El registro de vehículos sigue los siguientes requerimientos: se requiere almacenar marca, modelo y año. Una marca de vehículo define varios modelos. Para el caso de las marcas, se cuenta con catálogos en el que se almacena (clave y descripción). Para el caso de los modelos, solo se registra el nombre del modelo. Se guarda el número de serie del vehículo.
Se ha decidido realizar la siguiente clasificación: Vehículos para transporte público, vehículos de carga y vehículos particulares. Para los vehículos de carga se almacena la capacidad en toneladas, y dependiendo del modelo se puede expresar la capacidad en metros cúbicos. En caso de aplicar, se guarda el número de remolques que tiene.
Para el caso de los vehículos de transporte público se requiere almacenar el número de pasajeros sentados, pasajeros parados que puede transportar. Para tener un mayor control, se requiere almacenar el tipo de licencia que se requiere para poder conducir dicho automóvil: Tipo A licencia para taxis tipo sedán, tipo B, Licencia para camionetas que no pueden transportar pasajeros de pie, y Licencia tipo C para Autobuses y Camiones que tienen capacidad para transportar a 20 o más pasajeros.
Para los autos particulares se requiere almacenar las siguientes características del auto que son requeridas para cálculo de seguro e impuesto: número de bolsas de aire, bandera que indica si cuenta con frenos ABS, tipo de transmisión (M = Manual, A = Automática).
Se registran los datos del propietario del vehículo: nombre, apellidos, RFC (opcional), licencia(s) y correo electrónico (obligatorio). Un auto cuenta con un solo propietario a la vez.
¿Independiente o del vehículo?
¿En la BD o registro para verificación?
Para realizar el proceso de verificación se aplica la siguiente regla: A cada vehículo se le deberá instalar un dispositivo que mide y reporta las emisiones de contaminantes. Cada dispositivo cuenta con un número único de serie que se debe registrar al momento de registrar el vehículo. Este dispositivo se comunicará con el sistema central y reportará las siguientes cantidades de contaminantes Hidrocarburos = HC, Monóxido de Carbono CO en un rango de [0,1], Óxido de Nitrógeno NOx, Dioxido de Carbono CO2. El reporte de las emisiones se realiza cada 10 días. Se requiere guardar estas medidas y la fecha de registro.
para quién Rango?
check?
En el momento que el sistema detecta que el vehículo no cumple con los niveles permitidos, se genera una notificación que se le envía por correo electrónico. Se requiere registrar el número de notificación (1, 2 o 3), la fecha de envío y el registro de mediciones que envió el dispositivo que causaron el exceso.
¿Que significa cada uno?
Para llevar un control adecuado de los vehículos, la empresa propone los siguientes estados: EN REGLA, CON LICENCIA EXPIRADA, CON ADEUDO DE IMPUESTO, CON VERIFICACION PENDIENTE. Se requiere conocer en todo momento el estado actual del auto y la historia de cambio de estados a lo largo del tiempo.
Cuando el sistema genera notificaciones, el dueño del automóvil debe acudir a verificar su auto. Se registra los datos de la verificación fecha de verificación, folio de verificación (13 caracteres) y un nuevo registro de las emisiones contaminantes. Si el vehículo pasa la nueva medida, el status del auto se actualiza a EN REGLA, siempre y cuando no tenga otros adeudos. ?
A lo largo de la vida útil de un vehículo, este puede tener varios propietarios (uno a la vez). Se ha decidido guardar adicional al propietario actual, la lista de propietarios que ha tenido el automóvil a lo largo del tiempo. Se requiere guardar el periodo en el que cada propietario tuvo a su cargo dicho automóvil. Cada propietario debe contar con su licencia que tiene un periodo finito de vigencia. Se requiere almacenar los siguientes datos: foto, firma y huella de los índices del propietario, número de licencia, periodo de vigencia. Al término de la vigencia de la licencia, esta debe ser reemplazada por una nueva. Una vez que la nueva licencia ha sido generada, se debe asociar a la licencia que reemplaza (en caso de existir). Cabe mencionar que a un propietario se le permite tener hasta 3 licencias (una por cada tipo de licencia).
Para sostener el mantenimiento y operación del sistema de control de automóviles, a cada auto se le cobra una cuota anual. Se requiere registrar la fecha de pago, el importe y el folio de pago que se realiza por cada automóvil.
Con la finalidad de promover el uso adecuado del automóvil, un propietario puede recibir puntos negativos. Si el propietario acumula 20 o más puntos, perderá su derecho a contar con licencia. Se requiere almacenar la fecha en la que se registran los puntos negativos, descripción de la falta, cantidad de puntos negativos asignados y un documento PDF que contiene el detalle y la evidencia de la(s) faltas cometidas.
Finalmente, algunos automóviles pueden ser usados como autos de carga y particulares a la vez. Por ejemplo, una camioneta adaptada con un compartimiento trasero. Solo para esta combinación, el sistema permite que el mismo automóvil tenga ambos roles.

## Entidades candidatas:

- Cátalogo de entidades. Entidad. clave, nombre
- Cátalogo de placas. Placa. num_placa, entidad, fecha_asignación
- Vehículo marca, modelo, año, num_serie
- Marca clave, descripcion
- Modelo ? nombre
- V. público p_parados, p_sentados, licencia
  - taxis { 0 parados, 4 sentados
  - camionetas { n sentados, 0 parados
  - camión { sentados + parados ≥ 20
- V. carga capacidad_toneladas, capacidad_metros (null), num_remolques (null)
- V. particular num_bolsas_aire, frenos, transmison
- propietario nombre, apellidos, RFC (null), licencia(s), correo electrónico (check)
- licencia tipo (multi? M:N?*)
- * Dispositivo
  - Histórico de emisiones
  - Notificacion (dependencia de identificación)
- estatus vehiculo
  - historico estatus vehiculo
  - verificación. fecha_verificacion, folio,
  - Contaminantes ?
  - Histórico de propietarios
  - licencia_expirada
  - Cuota_anual
  - penalización

## Dudas

- Check de licencia
- Estado de placa (activa / inactiva)
- Medicion_contaminantes → ¿Representa todas las posibles combinaciones?

1_

2_

3_

4_

5_ Traslape total

*Tipo de licencia como atributo

6, 7.

* ¿Cómo se relaciona la verificación?

* ¿El histórico sería la tabla intermedia entre tipo_medición y dispositivo?

| medición_id | tipo_medic. | dispositivo_id | notificación_id |
|---|---|---|---|
| 1 | 1 | 100 | null |
| 2 | 2 | 100 | 500 |
| 3 | 3 | 100 | null |

?

sobrepasa

mismo coche

| emisión_id | notificación_id | num_notificación |
|---|---|---|
| 1 | null | null |
| 2 | 100 | 1 |
| 2 | 101 | 2 |

¿Cómo validamos para que se incremente?

8, 9, 10, 11, 12

Para las nuevas mediciones de verificación, ¿se debe enviar una notificación en caso de superar los límites? eso ya no se especifica en las reglas.

| medición_id | tipo_medic. | dispositivo_id | notificación_id |
|---|---|---|---|
| 1 | 1 | 100 | 500 |
| 2 | 2 | 100 | null |
| 3 | 4 | 300 | null |
| 4 | 1 | 500 | 500 |
| 5 | 2 | 800 | null |
| 6 | 3 | 600 | null |
| 7 | 2 | 300 | 300 |
| 8 | 2 | 600 | 500 |

¿Cómo sabes cual es la correcta?

13, 14.

15, 16,

17_

1:M
Vehiculo
(1,1)
pago
(1,n)
cuota_anual

18_

1:M
puntos negativos
(0,n)
se asignan
(1,1)
propietario

19_

1:M
v_publico
(1,n)
tiene
(1,1)
licencia_publica

Entidad
Clave
Nombre
tiene
1:M
placa
num_placa
Fecha_asignacion
estatus_vehiculo
descripcion
tiene
1:M
historico
M:N
Fecha_status
cuota_anual
Fecha_pago
Folio
importe
es pagada
le pertenece
1:1
vehiculo
num_serie
Fecha_status
año
Fecha_inicio
tiene
1:M
Modelo
nombre
pertenece
1:M
Marca
clave
descripcion
historico
M:N
Fecha_status
pertenece
1:M
Fecha_inicio
Fecha_fin
es_publico
es_particular
es_carga
V_particular
num bolsas_aire
frenos ABS
tiene
1:M
tipo_transmision
descripcion
clave
V_carga
cap_ton
cap_m3
num remolques
puede ser derivada
V_publico
pasajeros parados
pasajeros sentados
tiene
1:M
licencia_publico
clave
descripcion
puntos_negativos
Fecha
descripcion faltas
cantidad_puntos
PDF
puntos_id
se asignan
1:M
propietario
nombre
ap_paterno
ap_materno
RFC
correo
tiene
1:M
licencia
Foto
Firma
huella derecha
huella izquierda
num licencia
periodo
tiene
1:1
tiene
1:1
dispositivo
num_serie
Fecha registro
registra
M:N
Fecha registro
valor
envia
1:M
Notificacion
num notificacion
Fecha envio
genera
1:1
Verificacion
Folio
tipo_medicion
clave
nombre
es usado
M:N
valor
Fecha verificacion
Fecha verificacion

Entidad
Entidad_id
Clave (UK)
Nombre
Placa
Placa_id
Num_placa (UK)
Fecha_asignacion
Entidad_id. (FK)
Historico_estatus_vehiculo
Historico_estatus_vehiculo_id
Fecha_estatus
Estatus_vehiculo_id. (FK)
Vehiculo_id. (FK)
Cuota_anual
Cuota_anual_id
Folio (uk)
Importe
Fecha_pago
Vehiculo_id. (FK)
Vehículo
Vehiculo_id
Num_serie (UK)
Anio
Es_particular
Es_carga
Es_publico
Fecha_inicio
Fecha_estatus
Placa_id. (UK)(FK)
Modelo_id. (FK)
Propietario_id. (FK)
Dispositivo_id. (UK)(FK)
Estatus_vehiculo_id. (FK)
Estatus_vehiculo
Estatus_vehiculo_id
Descripción
Marca
Marca_id
Clave (uk )
Descripcion
Modelo
Modelo_id
Nombre
Marca_id (FK)
Puntos_negativos
Puntos_negativos_id
Fecha_puntos
Cantidad_puntos
PDF
Propietario_id. (FK)
Propietario
Propietario_id
Nombre
Ap_paterno
Ap_materno
RFC (uk) Null
Correo (uk)
Dispositivo
Dispositivo_id
Num_serie (uk)
Fecha_registro
Dispositivo_medicion
Dispositivo_medicion_id
Valor
Fecha_registro
Dispositivo_id. (FK)
Tipo_medicion_id (FK)
Notificacion_id. Null (FK)
Tipo_medicion
Tipo_medicion_id
Clave (uk)
Nombre
Descripcion_falta
Descripcion_falta_id.
Descripcion
Puntos_negativos_id (FK)
Historico_vehiculo_propietario
Historico_vehiculo_propietario_id
Fecha_inicio
Fecha_fin
Propietario_id. (FK)
Vehiculo_id. (FK)
Licencia
Licencia_id
Num_licencia. UK
Foto
Firma
Huella_der. Null
Huella_izq. Null
Perido_vigencia
Licencia_anterior_id Null.(FK)
Propietario_id. (FK)
(0,1)
Vehiculo_particular
Vehiculo_id
Num_bolsas_aire
Frenos_ABS
Tipo_Transmision_id. (FK)
Vehiculo_carga
Vehiculo_id
Cap_toneladas
Cap_m_cubico. Null
Num_remolques Null
Vehiculo_publico
Vehiculo_id
Num_parados
Num_sentados
Tipo_licencia_id. (FK)
Tipo_Transmisión
Tipo_Transmision_id
Clave. UK
Descripcion
Tipo_licencia
Tipo_licencia_id
Clave. UK
Descripcion
Verificación_medicion
Verificacion_medicion_id
Valor
Fecha_verificacion
Tipo_medicion_id. (FK)
Verificacion_id. (FK)
Notificación
Notificación_id
Num_notificacion
Fecha_envio
Verificación
Verificacion_id
Folio (uk)
Fecha_verificación
Notificacion_id (UK)(FK)

Entidad
Entidad_id
Clave (UK)
Nombre
No índice
Placa
Placa_id
Num_placa (UK)
Fecha_asignacion
Entidad_id. (FK)
Historico_estatus_vehiculo
Historico_estatus_vehiculo_id
Fecha_estatus
Estatus_vehiculo_id. (FK)
Vehiculo_id. (FK)
Cuota_anual
Cuota_anual_id
Folio (uk)
Importe
Fecha_pago
Vehiculo_id. (FK)
Vehículo
Vehiculo_id
Num_serie (UK)
Anio
Es_particular
Es_carga
Es_publico
Fecha_inicio
Fecha_estatus
Placa_id. (UK)(FK)
Modelo_id. (FK)
Propietario_id. (FK)
Dispositivo_id. (UK)(FK)
Estatus_vehiculo_id. (FK)
Estatus_vehiculo
Estatus_vehiculo_id
Clave
Marca
Marca_id
Clave (uk )
Descripcion
Modelo
Modelo_id
Nombre
Marca_id (FK)
Puntos_negativos
Puntos_negativos_id
Fecha_puntos
Cantidad_puntos
PDF
Propietario_id. (FK)
Propietario
Propietario_id
Nombre
Ap_paterno
Ap_materno
RFC (uk) Null
Correo (uk)
Descripcion_falta
Descripcion_falta_id.
Descripcion
Puntos_negativos_id (FK)
Historico_vehiculo_propietario
Historico_vehiculo_propietario_id
Fecha_inicio
Fecha_fin
Duracion_meses
Propietario_id. (FK)
Vehiculo_id. (FK)
Licencia
Licencia_id
Num_licencia. UK
Foto
Firma. Null
Huella_der. Null
Huella_izq. Null
Perido_vigencia
Licencia_anterior_id Null.(FK)
Propietario_id. (FK)
Vehiculo_particular
Vehiculo_id
Num_bolsas_aire
Frenos_ABS
Tipo_Transmision_id. (FK)
Vehiculo_carga
Vehiculo_id
Cap_toneladas
Cap_m_cubico. Null
Num_remolques Null
Vehiculo_publico
Vehiculo_id
Num_parados
Num_sentados
Tipo_licencia_id. (FK)
Tipo_Transmisión
Tipo_Transmision_id
Clave. UK
Descripcion
Tipo_licencia
Tipo_licencia_id
Clave. UK
Descripcion
Dispositivo
Dispositivo_id
Num_serie (uk)
Fecha_registro
Dispositivo_medicion
Dispositivo_medicion_id
Valor
Fecha_registro
Dispositivo_id. (FK)
Tipo_medicion_id (FK)
Notificacion_id. Null (FK)
Tipo_medicion
Tipo_medicion_id
Clave (uk)
Nombre
Verificación_medicion
Verificacion_medicion_id
Valor
Fecha_verificacion
Tipo_medicion_id. (FK)
Verificacion_id. (FK)
Notificación
Notificación_id
Num_notificacion
Fecha_envio
Verificación
Verificacion_id
Folio (uk)
Fecha_verificación
Notificacion_id (UK)FK)

Descripcion_falta

| Puntos_negativos_id (FK) |
| --- |
| Descripcion |

Puntos_negativos_id

| |
| --- |
| Fecha_puntos<br>Cantidad_puntos<br>PDF |

Modelo

| Modelo_id |
| --- |
| Nombre |

Marca

| Marca_id |
| --- |
| Clave (uk )<br>Descripcion |

Dispositivo

| Dispositivo_id |
| --- |
| Num_serie (uk)<br>Fecha_registro |

Tipo_medicion

| Tipo_medicion_id |
| --- |
| Clave (uk)<br>Nombre |

Vehículo

| Vehiculo_id |
| --- |
| Num_serie (uk)<br>Anio<br>Es_particular<br>Es_carga<br>Es_publico |

Propietario

| Propietario_id |
| --- |
| Nombre<br>Ap_paterno<br>Ap_materno<br>RFC (uk) Null<br>Correo (uk) |

Historico_vehiculo_propietario

| Historico_vehiculo_propietario_id |
| --- |
| Fecha_inicio<br>Fecha_fin |

Modelo

| Modelo_id |
| --- |
| Nombre |

Dispositivo_medicion

| Dispositivo_medicion_id |
| --- |
| Valor<br>Fecha_registro |

(0,1) (0,1) (0,1)

Vehiculo_particular

| Vehiculo_id |
| --- |
| Num_bolsas_aire<br>Frenos_ABS |

Vehiculo_carga

| Vehiculo_id |
| --- |
| Cap_toneladas<br>Cap_m_cubico. Null<br>Num_remolques Null |

Vehiculo_publico

| Vehiculo_id |
| --- |
| Num_parados<br>Num_sentados |

Verificación

| Verificacion_id |
| --- |
| Folio (uk)<br>Fecha_verificación |

Verificación_medicion

| Verificacion_medicion_id |
| --- |
| Valor<br>Fecha_verificacion |

Notificación

| Notificación_id |
| --- |
| Num_notificacion<br>Fecha_envio |

Estatus_vehiculo

| Estatus_vehiculo_id |
| --- |
| Descripción |

Historico_status_vehiculo

| Historico_status_vehiculo_id |
| --- |
| Fecha_status |

Licencia

| Licencia_id |
| --- |
| Num_licencia<br>Foto<br>Firma<br>Huella_derecha. Null<br>Huella_izquierda. Null<br>Perido_vigencia<br>Licencia_anterior_id Null |

Cuota_anual

| Cuota_anual_id |
| --- |
| Folio (uk)<br>Importe<br>Fecha_pago |

Puntos_negativos

| Puntos_negativos_id |
| --- |
| Fecha_puntos<br>Cantidad_puntos<br>PDF |

Transmisión

| Transmision_id |
| --- |
| Clave<br>Descripcion |

1_ ¿Varias mediciones (que sobrepasan) pueden generar la misma notificación o se tiene que generar una notificación por cada medición que exceda los valores permitidos?

2_ • Derivado
• Multivalorado
• cíclico
} diseño

## Tablas temporales

Reporte para los vehículos que están "en regla"

| Auto_Id | num_serie | propietario | Fecha_status |
|---|---|---|---|
| 15 | 45A32 | Luis | Hoy |
| 16 | 8467A | Fer | mañana |
| 4 | | Angel | |
| 6 | | | |
| 8 | | | |

```oracle
--Tabla para realizar reportes
create global temporary table reporte_vehiculo_temporal (
  vehiculo_id          number(10,0)  not null,
  num_serie            varchar2(40)  not null,
  propietario          varchar2(120) not null,
  estatus_vehiculo     varchar2(80)  not null,
  fecha_estatus        date          not null
)

```

Nota: Crear un procedimiento para generar el reporte utilizando la tabla temporal.

- externa correcta
- Muchas indices ?

## Actualiza_propietario

### Before

- Validar si es el mismo propietario
- Fecha fin en el historico

### After

- Actualizar fecha_inicio en vehicolo por sysdate
- Insertar en historico el nuevo propietari
-

**PROYECTO FINAL**
**ESPECIFICACIONES.**

## 1.1. INSTRUCCIONES GENERALES.

- El proyecto se puede realizar en equipos hasta de 2 personas.
- De forma opcional se puede desarrollar una aplicación web o standalone empleando el lenguaje y herramientas de su preferencia.
- No se requiere realizar reporte alguno. Solo se deberá presentar:
  - Hoja de asignación de proyecto con rúbrica. Deberá ser llenada con los datos de los integrantes del equipo.
  - Diagrama ER empleando Notación Chen (Opcional)
  - Modelo Relacional empleando notación Crow's Foot o IDEF1X.
- Durante la entrega del proyecto se deberá realizar una explicación y justificación de las actividades realizadas en cada uno de los scripts.

***Importante:***
Se hará una selección aleatoria para designar al integrante que realizará la explicación para cada script SQL desarrollado.

## 1.2. ACTIVIDADES A REALIZAR PARA EL CASO DE ESTUDIO ASIGNADO.

### *1.2.1. Construcción del modelo ER (Opcional)* √

El diagrama deberá contener todos los elementos vistos en las prácticas de modelado: cardinalidades, tipos de relación.

### *1.2.2. Construcción del modelo relacional.* √

El modelo deberá contener todos los elementos vistos en prácticas de modelado: cardinalidades, tipos de datos, restricción NULL/NOT NULL. La notación a emplear es libre: Crow's Foot o IDEF1X.

Tip: El caso de estudio asignado esta diseñado de tal forma que el modelo deberá contener los siguientes elementos:

- Atributos derivados
- Atributos multivalorados
- Dependencias de identificación
- Al menos una relación 1:1 y una relación M:N con atributos en la relación.
- Relaciones recursivas.
- Jerarquía de supertipo y subtipos.
- Manejo de datos con histórico.

### *1.2.3. Script s-01-usuarios.sql* √

Todos los Scripts SQL deberán contener el encabezado empleado en las prácticas:

```
--@Autor(es):        <Nombre de los integrantes>
--@Fecha creación:   dd/mm/aaaa
--@Descripción:      <breve descripción del contenido y propósito del archivo>
```

- Este script contendrá la definición de 2 usuarios: `<iniciales>_proy_invitado` y `<iniciales>_proy_admin`, donde `<iniciales>` corresponden a la primera letra de cada apellido paterno de cada integrante. El usuario `admin` será el dueño de todos los objetos del caso de estudio.
- Incluir en el script la definición de 2 roles:
  - `rol_admin` Contendrá todos los roles necesarios para poder implementar el caso de estudio asignado.
  - `rol_invitado`. Solo deberá tener permisos para crear sesiones.
- Asignar a los usuarios admin e invitado los roles `rol_admin` y `rol_invitado` respectivamente.

### *1.2.4. Script s-02-entidades.sql* √

- Este archivo contendrá el código DDL empleado para crear las tablas del caso de estudio. El código ***no deberá*** ser creado con ER-Studio. Todas las restricciones deberán tener un nombre asignado.
- Se deberá hacer uso de todos los tipos de restricciones vistas en clase: `unique, check, primary key, foreign key`.
- La definición de algunas tablas deberá incluir el uso de `default`, seleccionar al menos un caso donde pueda aplicarse.
- Seleccionar al menos un caso donde sea factible emplear una columna virtual. De ser posible se pueden crear nuevas tablas que hagan uso de este tipo de columnas.
- Sugerencias para su implementación:
    - Verificar la existencia de atributos derivados. De no existir se puede agregar algún campo para reproducir el escenario. Los atributos derivados son excelentes candidatos para crear tablas virtuales.

#### *1.2.5. Script s-03-tablas-temporales.sql*

- Diseñar al menos un escenario donde se haga uso de una o más tablas temporales. Se pueden crear nuevas tablas en caso de ser necesario.
- Sugerencias para su implementación:
    - Suponer que se desea consultar datos de un conjunto de tablas, pero por el nivel de normalización que presentan, se decide crear una tabla temporal para aplicar un proceso de desnormalización e insertar los datos de forma temporal. Por ejemplo, suponer que se tiene una Jerarquía Super tipo – Subtipos. Se decide crear una tabla temporal que contendrá todos los datos de la Jerarquía.
    - Simular un carrito de compras. El contenido del carrito de compras se puede guardar en una tabla temporal y en el momento que el cliente decida comprar, los datos se consultan de la tabla temporal y se insertan en las tablas permanentes.
    - Suponer que se tiene que realizar ciertos cálculos que provienen de la consulta de varias tablas permanentes como son: promedios, costos totales, descuentos, ventas, etc. El resultado de estos cálculos se puede guardar en una tabla temporal para ser consultados por algún cliente.

#### *1.2.6. Script s-04-tablas-externas.sql*

- Diseñar un escenario donde se haga uso de una o más tablas externas.
- Sugerencias:
    - Se tiene un archivo de texto y se desea consultar sus datos sin tener que cargar los datos al interior de la BD. El archivo de texto deberá contener una lista de registros separados por algún carácter, por ejemplo: “#”, “*”,”,”, etc.

#### *1.2.7. Script s-05-secuencias.sql*

- Este archivo contendrá la definición de ***todas*** las secuencias necesarias para poder insertar registros en tablas que requieran la generación de valores secuenciales. Todas las secuencias deberán iniciar en un número seleccionado por el equipo.

#### *1.2.8. Script s-06-índices.sql*

- Realizar un análisis y seleccionar algunos casos donde el uso de índices pudiera ser adecuado:
    - Uso de índices Non Unique para mejorar el desempeño de las consultas, por ejemplo, identificar posibles campos que son empleados frecuentemente en búsquedas (claves, números de cuenta, rfc, curp, email, etc). Considerar de forma adicional, indexar llaves foráneas que participan en operaciones JOIN frecuentes.
    - Uso de índices Unique empleados para verificar duplicidad de valores.
    - Uso de un índice compuestos tipo Unique, empleados para validar duplicidad de combinaciones de valores de las columnas involucradas en el índice.
    - Uso de uno o más índices basados en el uso de funciones. Por ejemplo, definir un índice con la función `lower` o `upper` que permita agilizar las búsquedas sin importar si los datos se encuentran capturados en minúsculas o mayúsculas.

#### *1.2.9. Script s-07-sinonimos.sql*

- Generar 3 o más sinónimos públicos que le pertenezcan al usuario admin. Los sinónimos deberán estar disponibles para que otros usuarios puedan emplearlos, por ejemplo, el usuario invitado.
- Generar las instrucciones necesarias para que el usuario admin otorgue permisos de lectura al usuario invitado en al menos 3 entidades.
- Generar 3 o más sinónimos que le pertenezcan al usuario invitado. Dichos sinónimos deberán ser empleados para leer el contenido de las tablas a las que el usuario admin le otorgó permisos de lectura.
- Finalmente, suponer que un software necesita que todas las tablas del proyecto tengan un prefijo formado por 2 caracteres: `XX_<nombre_tabla>`. Para implementar este requerimiento se creará un sinónimo privado para cada tabla. Se recomienda realizar un programa anónimo PL/SQL empleando SQL dinámico para evitar escribir manualmente los sinónimos.

#### *1.2.10. Script s-08-vistas-sql*

- Realizar un análisis y crear 3 o más vistas. Analizar posibles escenarios donde una vista podría ser adecuada.
- Posibles escenarios:
    - Generar una vista para ocultar la complejidad de la consulta en la que se involucran varias tablas: joins, funciones de agregación, etc.
    - Generar una vista para proteger el acceso a columnas consideradas como 'delicadas': contraseñas, números de tarjeta, etc. En este escenario, el usuario admin puede otorgar permisos al usuario invitado para que pueda leer el contenido de una vista que no contiene columnas privadas.

#### *1.2.11. Script s-09-carga-inicial.sql*

- Este script contendrá algunos datos de prueba y carga inicial para poder ilustrar el correcto funcionamiento de la base de datos.
- No existe límite en cuanto a la cantidad de registros, con unos cuantos registros es suficiente.
- El script deberá hacer uso de las secuencias para realizar las inserciones.
- Tener en cuenta que un valor generado por una secuencia pudiera emplearse varias veces para insertar registros con llaves foráneas. En este caso se recomienda emplear un Script PL/SQL y el uso de variables para reutilizar sus valores.

#### *1.2.12. Script s-10-consultas.sql*

- Este archivo contendrá 5 o más consultas. El criterio es libre. Se debe emplear el uso de los siguientes elementos.
    - joins (inner join, natural join, outer join)
    - funciones de agregación (`group by` y `having`)
    - algebra relacional (operadores set: `union, intersect, minus`).
    - Subconsultas
    - Consulta que involucre el uso de un sinónimo
    - Consulta que involucre el uso de una vista
    - Consulta que involucre una tabla temporal
    - Consulta que involucre a una tabla externa.
- No es necesario crear una consulta por cada elemento. En una misma consulta pueden incluirse varios de los elementos anteriores.

#### *1.2.13. Scripts s-11-tr-<nombre-trigger>.sql*

Empleando la nomenclatura de nombrado de estos scripts, generar los siguientes triggers.

- Al menos 2 triggers tipo row level. Posibles escenarios:
    - Validar reglas de negocio que no pueden ser implementadas por restricciones básicas como son unique, check. Si la regla de negocio no se cumple, el trigger puede lanzar una excepción para evitar que la operación DML se ejecute.
    - Realizar alguna operación DML sobre alguna tabla como consecuencia de la ocurrencia de un evento en otra, por ejemplo, actualizar el número de existencias de un producto cada vez que se confirma una orden de compra, actualizar el número de vacantes cada vez que se ocupa un lugar, etc.
    - Realizar la replicación de datos en tiempo real. Por ejemplo, aplicar las operaciones que se ejecutan en una tabla en otra tabla llamada tabla de respaldo o tabla de replicación.
    - No es necesario que en cada trigger se programen los 3 eventos: `insert , update y delete`. Estos 3 eventos pueden ser programados en diferentes triggers. La entrega se considera incompleta si en ninguno de los trigger se implementa alguno de estos 3 escenarios.
- Diseñar un escenario donde sea útil el uso de un compund trigger.
    - Tomar como referencia las prácticas complementarias y los apuntes del tema 10. Los escenarios típicos son: empleado para resolver el problema de tablas mutantes, y escenario para mejorar el desempeño con consultas que pueden actualizar grandes cantidades de datos.

- Cada trigger deberá ser incluido en un archivo SQL. No programar escenarios triviales como mandar mensajes a pantalla cuando ocurre un evento, u operaciones simples. Se recomienda validar los escenarios con el profesor.

#### *1.2.14. Scripts s-12-tr-<nombre-trigger>-prueba.sql*

- Para cada uno de los triggers creados anteriormente se deberá crear un script SQL o un programa PL/SQL que ejecute un escenario para validar el correcto funcionamiento del trigger. El script deberá ser lo más automatizado posible y evitar la intervención manual. Emplear como referencia los archivos de prueba empleados en el tema 10. El script deberá imprimir los mensajes o consultas necesarias en consola para validar su funcionamiento de forma visual.

#### *1.2.15. Scripts s-13-p-<nombre-procedimiento>.sql*

- Realizar un análisis y diseñar escenarios donde sea adecuado el uso de procedimientos almacenados. Generar 2 o más procedimientos
- Posibles escenarios:
  - Procedimiento que puede ser invocado por un usuario para implementar una funcionalidad del caso de estudio. Por ejemplo, para crear una nueva orden de compra. El procedimiento aceptará los parámetros necesarios para realizar la creación de la orden de compra. El procedimiento realiza varias inserciones en tablas diferentes. A nivel general, el procedimiento oculta todas las operaciones que se requieren para ejecutar la funcionalidad del caso de estudio que implica la ejecución de varias sentencias SQL.
  - Procesamiento de Datos. El procedimiento realiza una consulta empleando un cursor y por cada registro encontrado realiza alguna acción, por ejemplo, exportar datos a un archivo de texto, realizar alguna operación DML en otra tabla por cada registro obtenido por la consulta.
- No realizar procedimientos triviales como inserciones simples a tablas, o consultas sencillas a tablas. Se recomienda validar los escenarios con el profesor.

#### *1.2.16. Scripts s-14-p-<nombre-procedimiento>-prueba.sql*

- Para cada uno de los procedimientos creados se deberá generar un script de prueba que permita validar su funcionamiento. De forma similar a los scripts de prueba para triggers, se deberá evitar al máximo la intervención manual para verificar su funcionamiento.
- Estos programas deberán hacer un uso adecuado de las transacciones. Es decir, si el programa realiza operaciones DML, se deberá tener una estructura similar a la siguiente:

```
declare
  begin
    ----instrucciones, invocación de procedimientos, etc.
    ---commit al final de las operaciones, todo se ejecutó correctamente.
    commit;

  exception
    when others then
    --algo salio mal, se aplica rollback
    rollback;

end;
/
```

#### *1.2.17. Scripts s-15-fx-<nombre-funcion>.sql*

- Crear al menos 3 scripts, cada uno con la definición de una función. Algunos escenarios:
  - Realizar el calculo del total de una factura con base a los parámetros de entrada.
  - Procesar cierto número de parámetros para formar cadenas, por ejemplo, obtener un numero de serie a partir del tipo de auto, marca, modelo, etc.
  - Desarrollar funciones para leer archivos binarios de disco y regresar un objeto BLOB/CLOB para ser insertado en una tabla.

#### *1.2.18. Scripts s-16-fx-<nombre-funcion>-prueba.sql*

- Para probar las funciones del punto anterior, se puede realizar lo siguiente:
  - Crear un script o programa PL/SQL que la invoque para verificar su funcionamiento.
  - Emplear la función en algún otro programa (trigger, procedimiento, bloque anónimo). En este caso no será necesario crear un script especial para validar su funcionamiento.

#### *1.2.19. Scripts s-17-lob-<nombre-programa>.sql*

- El proyecto deberá incluir al menos un escenario donde se haga uso de datos LOB (columnas CLOB o BLOB). Se deberá generar un programa PL/SQL (procedimiento, trigger o programa anónimo) que maneje este tipo de dato. Posibles escenarios:
  - Suponer que en un directorio existen varias fotos de los empleados cuyo nombre de la imagen corresponde al id del empleado. A través de un programa PL/SQL se leerá cada foto y será actualizada en la base de datos.
  - Suponer que se tienen archivos de texto que corresponden a los reportes de los estudiantes, se desea que ese texto se almacene en la base de datos en una columna CLOB.
  - Crear una función que reciba como parámetro el nombre de un archivo binario. La función leerá el archivo binario y regresará un objeto tipo BLOB/CLOB. Esta función será empleada para realizar inserciones de datos BLOB/CLOB. Por ejemplo, suponer que se ha creado una función llamada `lee_foto`, una sentencia `insert` podría hacer uso de ella de la siguiente manera:

```
insert into empleado(empleado_id,foto)  values (1,lee_foto('mifoto.png'));
```

- Crear un trigger en el que cada vez que se haga la inserción o actualización de un dato BLOB/CLOB, su contenido sea copiado o exportado a un archivo en el servidor. El trigger emplearía el identificador o valor de la PK para ser asignado como nombre del archivo.

- No olvidar que este tipo de objetos requieren la creación de un objeto tipo `directory`.
- En caso que el modelo relacional asignado no contenga columnas BLOB/CLOB, agregar al menos una columna en algún objeto que pueda requerirlo, por ejemplo, la foto de un auto, el texto de un artículo, el pdf de un libro, el pdf que representa el comprobante de pago, etc.

#### *1.2.20. Scripts s-18-lob-<nombre-programa>-prueba.sql*

- Similar a los triggers y procedimientos, deberá existir un script de prueba que verifique el correcto manejo de objetos LOB. Para validar la correcta inserción, actualización o eliminación de estos objetos bastará con imprimir la longitud del objeto. Por ejemplo:

```
select empleado_id, dbms_length(foto) as longitud_foto
from empleado
where empleado_id = 1;
```

#### *1.2.21. Scripts s-19-cur-<nombre-programa>.sql*

- El proyecto deberá incluir el uso de uno o más cursores. El cursor puede aparecer en alguno de los procedimientos o triggers mencionados en los puntos anteriores, o si se desea, se puede crear un nuevo programa PL/SQL.

Notar que en un solo escenario se pueden implementar varios requerimientos: Por ejemplo, en un trigger se puede incluir el uso de una función creada por el alumno, el uso de cursores, o el manejo de datos BLOB. Lo mismo puede ocurrir para un procedimiento. Si esto ocurre ya no será necesario crear cada uno de los scripts antes mencionados.

### 1.3. Presentación y ejecución del proyecto.

Para realizar la demostración del correcto funcionamiento del proyecto, crear los siguientes archivos:

#### *1.3.1. Script s-00-main.sql*

- Este script deberá invocar a todos los scripts necesarios para realizar las siguientes acciones:
  - Eliminar al usuario en caso que exista
  - Crear todos los objetos necesarios.
  - Los triggers pueden ser omitidos o se pueden ejecutar hasta el final en caso de requerirse. Esto dependerá de los escenarios de prueba a verificar (Los triggers no deberán ser probados a través de la carga inicial, se deberá emplear los scripts de prueba).

- Antes de ejecutar el script principal, se deberá mostrar el contenido de cada script. Contestar las preguntas realizadas por el profesor.
- Posterior a la serie de preguntas, ejecutar el script.
- Se tomará en cuenta el correcto uso de usuarios, es decir, utilizar a los usuarios Oracle del sistema operativo y SYS de la base de datos solo para cuestiones administrativas.
- Ejecutar los scripts de prueba para validar, triggers, procedimientos, objetos LOBs, funciones y cursores.
- Finalmente ejecutar el script `resultados-proyecto-final.sql` que se encuentra en la carpeta compartida `BD/teoria/proyecto-final`

### 1.4. Aplicación Web/Standalone

- En caso de haber desarrollado aplicación, se deberá realizar un pequeño demo de su uso. No se requiere programar todo el caso de estudio. Basta con programar un caso de uso pequeño. Por ejemplo: Registrar artículo, registrar viaje, consultar reservación, etc.

# Diseño Conceptual Car - Control

DISEÑO LÓGICO CROWS FOOT