1. PROYECTO FINAL

Antes de comenzar con el desarrollo del presente documento es importante leerlo hasta el final, y evitar así actualizaciones o correcciones en pasos previos.

1.1. Descripción general

El proyecto final se compone por 2 partes:

1. Diseño lógico de un caso de estudio
2. Implementación de la base de datos

1.2. Diseño lógico del caso de estudio.

La primera actividad a desarrollar es la construcción de un modelo relacional para el caso de estudio asignado. Solicitar la asignación del caso de estudio en caso de requerirse. El diseño lógico de este caso de estudio deberá cumplir con los siguientes lineamientos:

- Leer cuidadosamente el caso de estudio. De forma opcional se puede construir un modelo ER empleando notación Chen. No se requiere presentar este modelo.
- El diseño lógico del caso de estudio deberá ser expresado a través de un modelo relacional empleando la herramienta ER Studio. La herramienta se puede obtener de la carpeta compartida `BD/practicas`.
- Antes de comenzar con la elaboración del modelo, la herramienta deberá ser configurada y personalizada con base al documento `BD/practicas/practica04/practica04-previo.pdf`.
- Desarrollar el modelo relacional empleando los conceptos aprendidos en el curso básico de Bases de Datos. Las reglas de negocio del caso requieren aplicar los siguientes conceptos de diseño:
    - Uso de notación Crow's foot.
    - Jerarquías de generalización y especialización (Tablas supertipo, subtipo y discriminantes). El modelo deberá indicar el tipo de Jerarquía: exclusión / traslape, parcial / total. No olvidar incluir la definición de su correspondiente discriminante.
    - Diseño de tablas con históricos.
    - Dependencia de identificación
    - Agregaciones
    - Atributos multivalorados
    - Relaciones recursivas
    - Uso de relaciones identificativas y no identificativas.
    - Indicar cardinalidades en cada relación empleando la notación `(Xmin, Xmax).`

    - Normalización del modelo por lo menos a nivel 3.
- Aplicar las siguientes convenciones y buenas prácticas en cuanto diseño de modelos relacionales:
    - Los nombres de las tablas deben representar entidades (a excepción de tablas intermedias). Su nombre debe ser un sustantivo en singular.
    - A medida de lo posible hacer uso de llaves primarias artificiales para obtener el mejor desempeño posible.
    - Emplear la convención `<NOMBRE_TABLA>_ID` para nombrar a los atributos que representan a una llave primaria artificial.
    - Evitar dejar espacios o huecos entre tablas, realizar la distribución de tablas en el modelo lo más compacto posible.
    - Evitar que las relaciones crucen otras tablas que no correspondan con la tabla destino.
- En caso de no estar familiarizados con los conceptos que se revisan en la asignatura Bases de Datos revisar las notas o videos correspondientes a los temas 04 y 05 del curso ubicados en `BD/teoría/tema04` y `tema05`.
- Al concluir el modelo se deberá solicitar una ***revisión*** en la que ambos integrantes deberán exponer y explicar el modelo obtenido. Parte de la calificación del proyecto se obtiene al realizar esta revisión. La implementación del proyecto no podrá continuar hasta no haber realizado esta revisión.

1.3. Creación de la base de datos.

- Comenzar con la creación de la base de datos una vez que se haya revisado y verificado el diseño lógico realizado en la sección anterior.
- Todas las configuraciones y código deberá ser guardado en scripts SQL o archivos shell. Emplear la notación `s-nn-<descripcion-corta>.sql` o `s-nn-<descripcion-corta>.sh`. `nn` se refiere al número de script iniciando en `01`, `<descripción-corta>` Representa a una cadena separada por guiones medios que indica una descripción corta en cuanto al propósito del script. Ejemplo: `s-01-creacion-bd.sql`
- Todos los scripts deberán contener su encabezado que incluye integrantes, fecha de creación, descripción del script.
- El código deberá estar correctamente formateado.
- Una vez que el proyecto haya sido concluido, generar una tabla con el resumen de todos los scripts creados

Nombre del script	Descripción

1.3.1. Simulación de dispositivos de almacenamiento

- La nueva base de datos deberá contar con una distribución de los datos en múltiples dispositivos de tal forma que represente una solución óptima aplicando los conceptos vistos durante el curso.
- Para simular la existencia de dispositivos de almacenamiento, se sugiere crear carpetas que simulan puntos de montaje. Se recomienda tomar como base la ruta `/unam-bda/d1<n>`. Por ejemplo, la nueva base de datos hará uso de 10 discos representados por los siguientes puntos de montaje (carpetas) `/unam/bda/d11`, . . . , `/unam/bda/d20`.
- No se recomienda emplear loop devices para almacenar data files, respaldos o el espacio asignado a la FRA. Loop devices pueden emplearse sólo para almacenar Redo Logs, archivos de control.

1.3.2. Configuraciones iniciales para crear la nueva base de datos.

- Crear una nueva base de datos empleando como nombre `<iniciales>proy`. `<Iniciales>` corresponden con los 2 primeros caracteres del apellido paterno de cada integrante. Si el proyecto se realiza de forma individual, emplear las primeras 2 letras del apellido paterno y las 2 primeras letras del apellido materno.
- Si el espacio en disco resulta ser un impedimento para crear esta nueva base de datos, eliminar la base de datos 1 creada al inicio del semestre. Se recomienda contar con 10 GB de espacio para evitar problemas durante el desarrollo del proyecto.

Proponer una configuración inicial y llenar la siguiente tabla:

Configuración	Descripción y/o configuración
Número y ubicación de los archivos de control.	
Propuesta de grupos de REDO	Un miembro de cada grupo deberá ubicarse en la FRA. No olvidar: Los data files no deberían ubicarse en los mismos discos donde se encuentran los Redo Logs y archivos de control.
Propuesta de juego de caracteres	
Tamaño del bloque de datos.	
Lista de parámetros que serán configurados al crear la base de datos.	Especificar nombre y valor.
Archivo de passwords	Indicar los usuarios que contendrá este archivo de forma inicial. Como requisito indispensable, deberá existir un usuario diferente a sys que será el encargado de realizar la administración de backups.

1.3.3. Módulos del sistema.

Con base a los requerimientos y a las características del caso de estudio, proponer una división por módulos funcionales. La idea es que estos módulos puedan ser administrados de forma independiente, en especial sus estructuras físicas de almacenamiento. Los datos de cada módulo deberán ser almacenados en tablespaces separados para poder implementar esta independencia de administración. Cada módulo deberá contar con un usuario dueño de todos los objetos. Se recomienda crear solo 2 módulos. Llenar la siguiente tabla.

Nombre del módulo	Descripción	Usuario

1.3.4. Esquemas por módulo.

Con base al modelo relacional realizado anteriormente realizar una distribución de los objetos considerando la propuesta de módulos realizada. Llenar la siguiente tabla.

Nombre de la tabla	Nombre del módulo

1.3.5. Esquema de indexado.

Con base a las reglas de negocio del caso de estudio asignado generar una lista de índices que serían considerados como necesarios para implementar reglas de negocio que requieran valores únicos, o para mejorar desempeño. Por ejemplo, indexar FKs, índices basados en funciones, etc. Llenar la siguiente tabla:

Módulo	Nombre de la tabla	Nombre del índice	tipo	Propósito
			unique, non unique, basado en funciones, lob index	

1.3.6. Diseño de tablespaces.

Con base a los requerimientos del caso de estudio, proponer un diseño físico en el que se describen los tablespaces que serán creados para soportar la operación de la base de datos para cada uno de los módulos. Especificar tanto los tablespace requeridos y recomendados así como los tablespaces creados para almacenar los datos del módulo.
Sugerencias:

- Considerar el almacenamiento de objetos CLOB, BLOB en discos diferentes. Considerar el uso de Big File tablespaces.
- Realizar un análisis y determinar las tablas que pudieran tener una gran cantidad de registros con el paso del tiempo, proponer un esquema simple de particionamiento. Considerar almacenar los datos de cada partición en un disco diferente.

- Considerar el almacenamiento de los índices en un tablespace separado.
- No olvidar que la distribución propuesta debe cuidar en todo momento que si los tablespaces de un módulo se detienen o fallan, el otro módulo debería seguir operando sin mayor problema.
- Tener presente y procurar en todo momento evitar posibles problemas de contención, por ejemplo, considerar crear tablespaces con más de un data file donde cada archivo se almacena en discos diferentes.

1.3.6.1. Definición de tablespaces comunes a los módulos

- En esta tabla se documentan los tablespaces comunes a los módulos.

Nombre del tablespace	Configuración
	Especificar: Big File o múltiple data files, tamaño, tipo de administración de segmentos y extensiones, ubicación de sus data files.

1.3.6.2. Definición de tablespaces por módulo

- Llenar la siguiente tabla con el diseño propuesto de tablespaces.

Módulo	Nombre del tablespace	Objetivo / Beneficio	Configuración
			Especificar: Big File o múltiple data files, tamaño, tipo de administración de segmentos y extensiones, ubicación de sus data files.

1.3.6.3. Asignación de tablespace por objeto y módulo

- Con base al diseño de tablespaces propuesto, cada módulo estará formado por varios tablespaces. En cada módulo existirán segmentos de diferente tipo (tablas, índices, particiones, objetos blob /clob) que requieren la asignación de su correspondiente tablespace. En esta tabla se podrá realizar una propuesta de distribución de los diferentes segmentos empleando los tablespaces definidos anteriormente.

Módulo	TIpo de segmento	Nombre del segmento	Nombre del tablespace
	(índice, tabla, partición, objeto clob, objeto blob)		

1.3.7. Creación de usuarios.

Con base a los usuarios propuestos anteriormente, generar las sentencias SQL necesarias para realizar su creación. Importante: Los usuarios solo deberán contar con los privilegios mínimos indispensables. No emplear sentencias SQL que otorguen privilegios generales, no hacer uso de `any`, `admin option`, etc. Cada uno de estos usuarios deberá tener cuota ilimitada en los tablespaces donde se almacenarán sus objetos. Su tablespace por default será el tablespace donde se almacenarán las tablas de su respectivo módulo.

Llenar las siguientes tablas:

Nombre del usuario	Default tablespace	Default temporary tablespace	Default undo tablespace	Lista de privilegios asignados al usuario
	(Colocar aquí el nombre del tablespace)	(Colocar aquí el nombre del tablespace)	(Colocar aquí el nombre del tablespace)	

1.3.8. Generación del código DDL para el modelo relacional.

A partir del modelo relacional generado anteriormente, realizar las siguientes acciones en ER -Studio

- Crear un nuevo modelo lógico por cada uno de los módulos propuestos anteriormente.
- Incluir en cada modelo lógico las tablas que le corresponden.
- crear un modelo físico a partir del modelo lógico para Oracle a partir de cada uno de los módulos (modelos lógicos creados en el punto anterior).
- Los constraints deben ser creados como parte de la instrucción `create table`. Evitar el uso de `alter table` para crear constraints.
- Emplear las siguientes convenciones para realizar el nombrado de los constraints. Si el nombre es demasiado largo, pueden aplicar algunas abreviaturas que sean lo más claras posible.

Tipo Constraint	Convención de nombrado
unique	`<nombre_tabla>_<nombre_columna>_uk`
primary key	`<nombre_tabla>_pk`
references, foreign key	`<nombre_tabla_hija>_<nombre_columna>_fk`
check	`<nombre_tabla>_<nombre_columna>_chk`

- Revisar el documento `BD/practicas/practica04/practica04-previo.pdf` para configurar la herramienta e implementar fácilmente estos requerimientos.
- Generar el código SQL empleando ER-Studio.
- Editar el script generado para realizar las asignaciones de tablespaces tanto de tablas como para índices, PKs, índices tipo LOB.

1.3.9.Modos de conexión.

- Realizar las configuraciones necesarias de tal forma que un usuario pueda conectarse a la instancia ya sea en modo compartido o en modo dedicado o a través de un pool de conexiones. La configuración por defecto deberá ser modo dedicado.

1.3.10. Habilitar la FRA.

- Habilitar la FRA, realizar un cálculo estimado de su tamaño con base a la cantidad de datos que se pretenden almacenar (ver siguientes secciones).
- Una de las copias del archivo de control deberá almacenarse en la FRA
- Uno de los grupos de Online Redo Logs deberá ubicarse en la FRA.
- Una las copias de los Archive Redo Logs deberá ubicarse en la FRA
- Habilitar el uso de Flashback logs, proponer un periodo de retención de estos logs para garantizar las funcionalidades asociadas con el uso de flashback.
- Todos los respaldos deberán almacenarse en la FRA. (Archivos de control, archivos de parámetros, backups de tipo BackupSet, backups del tipo Copy).
- Tip: La FRA debería ubicarse en un disco separado lo suficientemente grande para almacenar todos estos objetos.

1.3.11. Modo archivelog

- La base de datos deberá estar en modo `archivelog`.
- Proponer 2 ubicaciones. Una de ellas deberá almacenarse en la FRA.
- Tip: Los discos donde se almacene la copia que no está en la FRA debería ser dedicado.

1.3.12. Planeación del esquema de respaldos.

Diseñar la estrategia que se empleará para realizar los respaldos de la base de datos. Esta estrategia deberá incluir:

- Tipos de backups a realizar
- Frecuencia de repetición
- Ubicaciones de respaldo (FRA)
- Política de retención de backups.
- Tamaño total en espacio en disco disponible para realizar backups.

Una vez que esta estrategia ha sido decidida y documentada, crear un script que genere al menos un respaldo para cada etapa o ciclo definido con base a la política de retención.

1.3.13. Carga de datos.

- Se recomienda emplear un generador de datos para realizar esta actividad. Tratar de poblar la base de datos con una buena cantidad de datos para similar un ambiente real. Para las tablas que potencialmente pudieran tener grandes cantidades de registros, se recomienda generar por lo menos 2000 registros.

- No es necesario generar la misma cantidad de datos BLOB, pueden ser solo unos cuantos.
- Debido a que se trata de la carga inicial, no se requiere generar datos REDO. Deshabilitar la generación de REDO mientras se realiza la carga inicial.
- Posterior a la ejecución de la carga inicial habilitar nuevamente la generación de REDO.

1.3.14. Respaldo inicial.

Una vez que los datos hayan sido cargados, ejecutar una primera iteración de la estrategia de respaldos programada. Revisar el espacio requerido

1.3.15. Simulación de la carga diaria.

- Generar Scripts que simulen la generación de datos de REDO los cuales representarán la carga diaria de una base productiva. Se recomienda tomar como base los scripts proporcionados en temas anteriores. Como mínimo se deberán generar aproximadamente 30 MB de datos REDO. Este valor también deberá ser considerado para decidir el tamaño de los grupos de REDO al momento de crear la base de datos.
- Realizar algunos ciclos de generación de datos REDO, y posteriormente hacer respaldos para comprobar su correcto funcionamiento.
- Ejecutar los comandos necesarios para liberar espacio en disco considerando archivos obsoletos.
- Llenar la siguiente tabla:

Programación de respaldos.

Fecha y hora	Datos REDO producidos (MB)	Fecha de Respaldo	Tipo de backup	Espacio requerido por el backup

1.3.16. Simular un proceso de Instance recovery.

- Ejecutar una simulación de datos de REDO de varios días.
- Hacer un `shutdown abort` y posteriormente iniciar para provocar un instance recovery.
- Revisar el tiempo requerido que se ocupó para realizar la recuperación de los datos.
- Tratar de ajustar el valor del parámetro `fast_start_mttr_target` de tal forma que el tiempo requerido para recuperar disminuya.

1.3.17. Simular un proceso de complete media recovery

- Dañar o eliminar uno de los data files de los módulos anteriores.
- Realizar un proceso de complete media recovery tanto manual como a través del uso del DRA.

1.3.18. Entrega del proyecto

- No será necesario generar reporte. Únicamente se deberá presentar los modelos relacionales.
- El día de la entrega los integrantes del equipo deberán explicar el contenido del proyecto y se les solicitará hacer algunas consultas en el diccionario de datos para validar los resultados empleando SQL Developer.