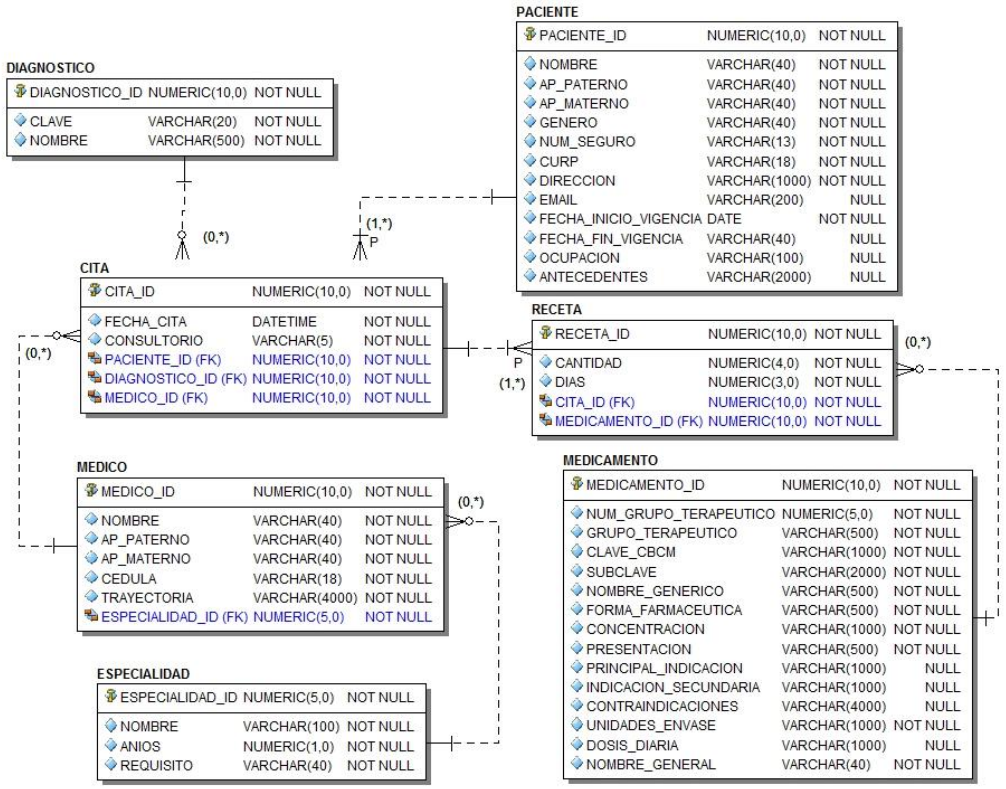
TEMA 5. OPTIMIZACIÓN DE CONSULTAS DISTRIBUIDAS
SERIE DE EJERCICIOS - PARTE 2

Nota: Los ejercicios se entregan de forma individual en cualquier formato.

Para los siguientes ejercicios considere nuevamente el caso de estudio del registro de citas médicas.



- El código DDL y DML pueden descargarse de la carpeta compartida BDD. Se puede emplear para verificar resultados. Importante: Ejecutar la recolección de estadísticas sobre el esquema que contiene a estas tablas para obtener buenos resultados.

1.1. VISUALIZACIÓN DE PLANES DE EJECUCIÓN

1.1.1. Ejercicio 1

Para la siguiente consulta:

- Obtener todos los datos de las citas donde el identificador del médico es el 10, o el identificador del paciente es 10.
- Considerar que existe un índice de tipo non unique en el campo `medico_id`

A. Generar el código SQL para definir los índices requeridos.
B. Generar el código SQL necesario para generar el plan de ejecución
C. Generar el código SQL para desplegar el plan de ejecución e incluir la salida. Proporcionar una explicación del plan generado.

1.1.2. Ejercicio 2

Para la siguiente consulta:

- Obtener todos los datos de las citas donde el identificador del médico es el 10.
- Considerar el mismo índice del ejercicio anterior.

A. Generar el código SQL necesario para generar el plan de ejecución
B. Generar el código SQL para desplegar el plan de ejecución e incluir la salida. Proporcionar una explicación del plan generado.

1.1.3. Ejercicio 3.

Para la siguiente consulta:

- Obtener el identificador y en nombre de todos los pacientes cuyo nombre inicie con la letra A.
- Existe un índice non-unique en el campo nombre.

A. Generar el código SQL para definir los índices requeridos.
B. Generar el código SQL necesario para generar el plan de ejecución.
C. Generar el código SQL para desplegar el plan de ejecución e incluir la salida. Proporcionar una explicación del plan generado.

1.2. ANÁLISIS DE PLANES DE EJECUCIÓN.

1.2.1. Ejercicio 1

- Considerar la siguiente información. Contestar las preguntas que se muestran al final.
- Índices:

```
create index cita_medico_ix on cita(medico_id);
create index medico_cedula_uk on medico(cedula);
```

- Consulta SQL.

```
explain plan for
select m.*, c.fecha_cita, c.consultorio
from medico m, cita c
where m.medico_id = c.medico_id
and m.cedula ='OAAAE344345WXXR';
```

- Plan de ejecución

```
-------------------------------------------------------------------
| Id  | Operation                           | Name             |
-------------------------------------------------------------------
|   0 | SELECT STATEMENT                    |                  |
|   1 |  NESTED LOOPS                       |                  |
|   2 |   TABLE ACCESS BY INDEX ROWID       | MEDICO           |
|*  3 |    INDEX UNIQUE SCAN                | MEDICO_CEDULA_UK |
|   4 |   TABLE ACCESS BY INDEX ROWID BATCHED| CITA             |
|*  5 |    INDEX RANGE SCAN                 | CITA_MEDICO_IX   |
-------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

   3 - access("M"."CEDULA"='OAAAE344345WXXR')
   5 - access("M"."MEDICO_ID"="C"."MEDICO_ID")
```

- Muestra de datos (Otras columnas se omiten por simplicidad).

MEDICO

Row_id	Médico_id	cedula
0A	1	OAAAE344345WXXN
0B	2	OAAAE344345WXXM
0C	3	**OAAAE344345WXXR**
0D	4	OAAAE344345WXXS
0E	5	OAAAE344345WXXT
0F	6	OAAAE344345WXXU

CITA

Row_id	Cita_id	Medico_id (fk)
1A	10	1
1B	20	2
1C	30	2
1D	40	3
1E	50	3
1F	60	3
1G	70	4

- Preguntas:

A. ¿Qué operación del plan de ejecución es la primera en ejecutarse?
B. Considerando la muestra de datos, dibujar una tabla que contenga los datos de la tabla outer del Nested Loop.
C. ¿Qué fuente de datos representa a la tabla Inner del Nested Loop?

D. ¿Cuántos accesos se realizarán a la tabla inner del Nested Loop ? o de forma similar: ¿Cuántas iteraciones se realizarán en el Nested Loop para recorrer a cada renglón de la tabla outer ?
E. Escriba la expresión booleana que se aplica en cada iteración del nested loop para ser aplicada a la tabla inner.
F. Escriba la expresión booleana que se aplica en la primera iteración del nested loop para ser aplicada a la tabla inner.
G. Escriba el resultado que se obtiene al ejecutar el paso 5 del plan de ejecución (iteración 1 del Nested Loop).
H. Dibuje una tabla con el resultado de ejecutar el paso 1 del plan de ejecución.

1.2.2. Ejercicio 2

- Considerar la siguiente información. Contestar las preguntas que se muestran al final.
- Consulta SQL:

```
explain plan  for
select r.receta_id,me.medicamento_id,me.nombre_general,r.cantidad
from medicamento me, receta r, cita c
where me.medicamento_id = r.medicamento_id
and r.cita_id = c.cita_id
and  to_char(c.fecha_cita,'yyyy') >='2016';
```

- Plan de ejecución

```
-----------------------------------------
| Id  | Operation           | Name        |
-----------------------------------------
|   0 | SELECT STATEMENT    |             |
|*  1 |  HASH JOIN          |             |
|*  2 |   HASH JOIN         |             |
|*  3 |    TABLE ACCESS FULL| CITA        |
|   4 |    TABLE ACCESS FULL| RECETA      |
|   5 |   TABLE ACCESS FULL | MEDICAMENTO |
-----------------------------------------
```

- Considerar la siguiente muestra de datos. La consulta real se aplicó con una cantidad mucho mayor de registros para provocar el uso de un Hash Join. Para efectos del ejercicio solo considerar los siguientes datos:

MEDICAMENTO

Row_id	Medicamento_id	Nombre_general
0A	1	Aspirina
0B	2	Paracetamol
0C	3	Mimepris
0D	4	Penicilina
0E	5	Morfina
0F	6	Acetol

RECETA

Row_id	Receta_id	cantidad	Medicamento_id	Cita_id
1A	100	4	1	40
1B	200	3	2	40
1C	300	5	2	30
1D	400	2	4	30
1E	500	4	4	20
1F	600	2	4	20

CITA

Row_id	Cita_id	Fecha_cita
2A	10	05/05/2016
2B	20	05/06/2017
2C	30	21/09/2015
2D	40	31/11/2012

A. ¿Qué operación es la que se ejecuta primero?
B. ¿Qué tabla se emplea para construir el primer Hash Table?
C. Genere una lista de parejas {(hash(<clave>),<valor>)} que contendrá el primer Hash Table, mencionar la columna que se emplea para formar la llave de cada elemento del Hash Table.
D. Escriba las N expresiones buscaEnHT(hash(<key>)) que se ejecutarán para poder satisfacer el primer join. Para cada expresión indicar el resultado que se obtendría. Indicar que campo se emplea para realizar la búsqueda

E. Dibuje una tabla de datos con el resultado del primer Join realizado a través de un Hash Table
F. Genere una lista de parejas `{(hash(<clave>),<valor>)}` que contendrá el segundo Hash Table, mencionar la columna que se emplea para formar la llave de cada elemento del Hash Table.
G. Escriba las N expresiones `buscaEnHT(hash(<key>))` que se ejecutarán para poder satisfacer el segundo join. Para cada expresión indicar el resultado que se obtendría. Indicar que campo se emplea para realizar la búsqueda
H. Dibuje una tabla de datos con el resultado del primer Join realizado a través de un Hash Table
I. Empleando la tabla anterior, mostrar la tabla final solicitada por la consulta.

1.3. CONSTRUCCIÓN DE PLANES DE EJECUCIÓN.

Para cada una de las siguientes consultas:
A. Generar una sentencia SQL que obtenga los datos solicitados.
B. Generar una propuesta de plan de ejecución que podría ser el más adecuado considerando las condiciones indicadas. Tomar el primer ejemplo resuelto como base sin considerar costos. Como apoyo adicional, es válido generar el plan de ejecución empleando el manejador para verificar resultados.
C. Generar una explicación del plan propuesto.

1.3.1. Ejercicio 1 (resuelto):

- Mostrar el nombre del médico y la fecha de sus citas de todos los médicos registrados en la BD.
- Forzar el uso de un Nested LOOP.
- Considerar que existe el índice en la FK (cita.medico_id) llamado cita_medico_ix

Respuesta:

A. Consulta SQL

```
explain plan for
select /*+ ORDERED USE_NL(c) */  m.nombre, c.fecha_cita
From  medico m, cita c
Where  m.medico_id =c.medico_id;
```

B. Plan de ejecución

id				Row source
1	Nested loop			
2		Nested loop		
3			Table acces full	Medico
4			Index range scan	Cita_medico_ix
5		Table Access by index row id		Cita

C. Explicación:
- El Hint especificado en la cláusula `select` obliga a ejecutar un Nested Loop (NL) empleando al alias 'c' como tabla inner.
- La primera operación a ejecutar es id=3. Se obtienen todos los registros de medico al no existir filtro alguno. Se elige a medico como outer por tener menor cardinalidad que cita.
- Por cada registro que se obtiene del paso anterior, se ejecuta id = 4 para obtener todas las citas de un determinado médico empleando un escaneo por rango del índice asociado a la FK. En resumen: ***por cada e.medico_id se buscan correspondencias con c.medico_id***
- Del paso anterior se obtiene una lista de ROW_IDs de los médicos que resultaron de la operación JOIN.
- En el último nested loop, se emplea como outer a la lista de row_ids y por cada uno se accede a la tabla cita para extraer el nombre de la cita (id = 5)

1.3.2. Ejercicio 2

- Obtener el nombre del paciente con curp 'ABC798701GYTOS7029'
- Considerar los siguientes índices:

```
create unique index paciente_curp_uk  on paciente(curp);
create index paciente_nombre_ix on paciente(nombre);
```

1.3.3. Ejercicio 3.

- Generar una propuesta de plan de ejecución para la consulta del ejercicio 1 considerando los siguientes cambios:
- El índice `cita_medico_ix` se ha eliminado
- La consulta ahora solicita mostrar el nombre del médico y el identificador (cita_id) de sus citas

1.3.4. Ejercicio 4

- Mostrar el nombre de los médicos cuya especialidad Inicie con “Neuro”
- Asumir que se han definido los siguientes índices:

```
create index especialidad_nombre_ix on especialidad(nombre);
create index medico_especialidad_ix on medico(especialidad_id);
```

- Asumir que se decide emplear un Nested Loop debido a que la cardinalidad de las tablas médico y especialidad es 5000 y 53 respectivamente.

1.3.5. Ejercicio 5

- Obtener todos los datos de los pacientes que tuvieron citas después del 2010
- Asumir que el manejador selecciona Hash Join como técnica ganadora.
- Asumir la existencia de los siguientes índices:

```
create index  cita_paciente_ix on cita(paciente_id);
```

- Las estadísticas indican que existen 2186 citas posterior al 2010
- Se estima que existen 15000 pacientes registrados.

1.3.6. Ejercicio 6.

- Obtener la CURP de todos aquellos pacientes que capturaron un email en el dominio @unam.mx
- Considerar los siguientes índices:

```
create index paciente_email_ix on paciente(email);
create unique index paciente_curp_uk on paciente(curp);
```

- El manejador decide realizar Hash Join (de requerirse).
- Al ser un índice de tipo unique para la CURP, se prefiere emplear este campo para construir un HashTable.

1.3.7. Ejercicio 7 (Resuelto).

- Obtener el nombre de los médicos que han recetado el medicamento con subclave '010.000.0091.00'
- Considerar los siguientes índices:

```
create unique index medicamento_subclave on medicamento(subclave);
create index  receta_medicamento_ix on receta(medicamento_id);
create index  receta_cita_ix on receta(cita_id);
create index  cita_medico_ix on cita(medico_id);
create index  medico_especialidad_ix on medico(especialidad_id);
```

Respuesta:

A. Sentencia SQL

```
explain plan for
select m.nombre
from medicamento me, receta r, cita c, medico m
where me.medicamento_id = r.medicamento_id
and r.cita_id = c.cita_id
and c.medico_id = m.medico_id
and me.subclave = '010.000.0091.00';
```

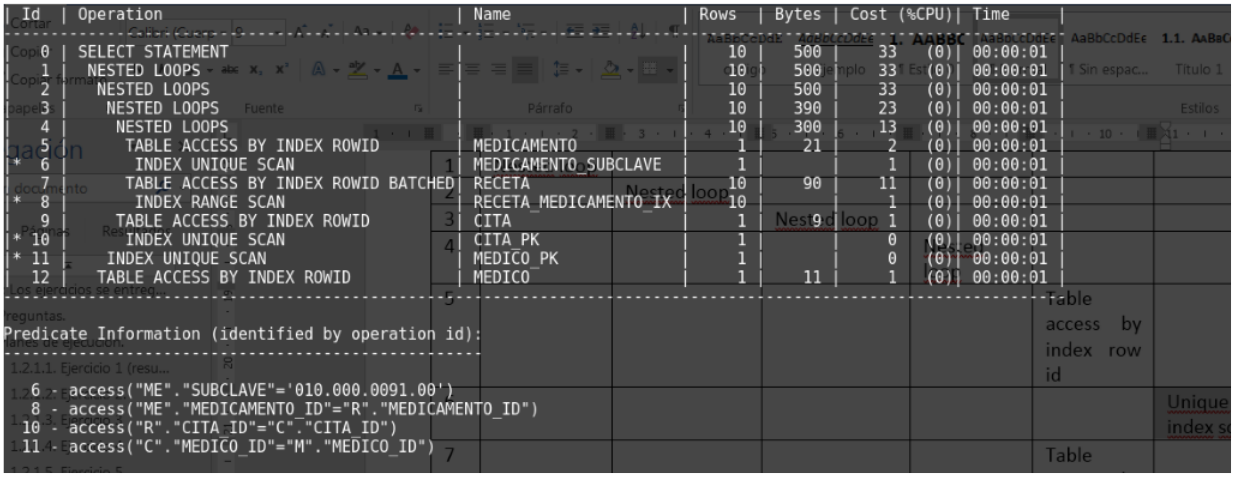
B. Plan de ejecución.

id							Row source
1	Nested loop						
2		Nested loop					
3			Nested loop				

4				Nested loop			
5					Table access by index row id		medicamento
6						Unique index scan	Medicamento_subclave_uk
7					Table access by index row id		receta
8						index range scan	Receta_medicamento_ix
9				Table access by index row id			Cita
10					Unique index scan		Cita_pk
11			Index unique scan				Medico_pk
12		Table access by index row id					medico

C. Explicación

- Se obtiene los datos de la tabla outer del primer Nested Loop. El Primer paso a ejecutar es el Num. 6, se obtiene 1 registro empleando el índice `Medicamento_subclave_uk`
- Segundo paso es el Num. 5, se obtiene el registro asociado al row_id anterior para poder recuperar `medicamento_id`
- Se obtiene la tabla inner para cada valor de la tabla outer. En este caso solo se tiene 1 registro. Se ejecuta en el paso 8 `index range scan` empleando el índice de la FK `receta_medicamento_ix`. En resumen, se aplica el match en la tabla inner `me.medicamento_id = r.medicamento_id`
- Con los ROW_IDs obtenidos, en el paso 8 se obtienen los registros de `receta`, principalmente para obtener `cita_id`.
- El resultado de este primer nested loop es la tabla outer del segundo NL (paso 4).
- La tabla inner del segundo NL es el resultado de ejecutar el paso 9. En el paso 10 se hace un unique index scan con la PK de cita para hacer el join `r.cita_id(fk) = c.cita_id(pk)`
- En el paso 9 se accede a CITA, básicamente para obtener `medico_id` para el siguiente join. Este resultado forma la tabla outer del tercer nested loop.
- La tabla inner del tercer NL es el resultado del paso 11. Se hace el match `c.medico_id = m.medico_id`, por ello se lee el índice `medico_pk`. Con esto se obtienen los ROW_IDs de los médicos solicitados. Solo falta obtener el nombre para lo cual se ejecuta el cuarto NL.
- En el último NL, la tabla inner es el paso 11. Se obtienen todos los registros de Medico empleando table Access by index row_id y así recuperar el nombre del médico.



```
| Id  | Operation                              | Name                | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                       |                     |    10 |   500 |    33   (0)| 00:00:01 |
|   1 |  NESTED LOOPS                          |                     |    10 |   500 |    33   (0)| 00:00:01 |
|   2 |   NESTED LOOPS                         |                     |    10 |   500 |    33   (0)| 00:00:01 |
|   3 |    NESTED LOOPS                        |                     |    10 |   390 |    23   (0)| 00:00:01 |
|   4 |     NESTED LOOPS                       |                     |    10 |   300 |    13   (0)| 00:00:01 |
|   5 |      TABLE ACCESS BY INDEX ROWID       | MEDICAMENTO         |     1 |    21 |     2   (0)| 00:00:01 |
|*  6 |       INDEX UNIQUE SCAN                | MEDICAMENTO_SUBCLAVE|     1 |       |     1   (0)| 00:00:01 |
|   7 |      TABLE ACCESS BY INDEX ROWID BATCHED| RECETA              |    10 |    90 |    11   (0)| 00:00:01 |
|*  8 |       INDEX RANGE SCAN                 | RECETA_MEDICAMENTO_IX|    10 |       |     1   (0)| 00:00:01 |
|   9 |     TABLE ACCESS BY INDEX ROWID        | CITA                |     1 |       |     1   (0)| 00:00:01 |
|* 10 |      INDEX UNIQUE SCAN                 | CITA_PK             |     1 |       |     0   (0)| 00:00:01 |
|* 11 |    INDEX UNIQUE SCAN                   | MEDICO_PK           |     1 |       |     0   (0)| 00:00:01 |
|  12 |   TABLE ACCESS BY INDEX ROWID          | MEDICO              |     1 |    11 |     1   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

   6 - access("ME"."SUBCLAVE"='010.000.0091.00')
   8 - access("ME"."MEDICAMENTO_ID"="R"."MEDICAMENTO_ID")
  10 - access("R"."CITA_ID"="C"."CITA_ID")
  11 - access("C"."MEDICO_ID"="M"."MEDICO_ID")
```