<u>Actividad 1:</u> En el cuaderno de prácticas realizar los ejercicios 3.71 a 3.75 referentes a la base de datos de jugadores de baloncesto:

- SQL:

3.71) Para cada equipo muestra el número de encuentros que ha disputado como local.

select nombreE, count(*) from equipos, encuentros where code=elocal group by nombreE order by nombreE;

3.72) Muestra los encuentros en los que se alcanzó mayor diferencia de puntos:

select equipos.nombreE, eq.nombreE, abs(PLocal - PVisitante) from equipos, equipos eq, encuentros where equipos.codE=ELocal and eq.codE=EVisitante and abs(PLocal - PVisitante)=

(select max(abs(encuentros.Plocal-encuentros.PVisitante)) from encuentros);

3.73) Muestra los jugadores que no han superado 3 faltas acumuladas:

select nombreJ, sum(faltas) from jugadores, alinieaciones where alineaciones.codJ=jugadores.codJ group by nombreJ having sum(faltas)<=3;

3.74) Muestra los equipos con mayores puntuaciones en los partidos jugados fuera de casa:

select EVisitante, PVisitante from encuentros where PVisitante=

(select max(encuentros.PVisitante) from encuentros);

3.75) Muestra todas las victorias de cada equipo, jugando como local o como visitante:

select nombreE, codE, ELocal, EVisitante from equipos, encuentros where (codE=ELocal and PLocal>PVisitante) or (codE=EVisitante and PVisitante>PLocal) order by nombreE;

<u>Actividad 2:</u> En el cuaderno de prácticas realizar los ejercicios 4.1, 4.2 y 4.3:

4.1) Crear una vista con los proveedores de Londres. ¿Qué sucede si insertamos en dicha vista la tupla ('S7', 'Jose Suarez', 3, 'Granada')?

```
create view vistal as select * from proveedor where ciudad='Londres';
insert into vistal values ('S7','Jose Suarez',3,'Granada');

Salida de Script *

Tarea terminada en 0,9 segundos
```

View VISTAl creado.

l fila insertadas.

Se inserta correctamente en la tabla proveedor a través de la vista1 dado que no existía ninguna tupla con la misma llave primaria en la tabla proveedor, pero como es lógico, no muestra la tupla en la tabla de la vista1 dado que no es de Londres.



4.2) Crear una vista con los nombres de los proveedores y sus ciudades. Inserta sobre ella una fila y explica cuál es el problema que se plantea. ¿Habría problemas de actualización?

```
create view vista2 as select nompro, ciudad from proveedor;

insert into vista2 values('Antonio Lopez', 'Madrid');

Salida de Script ×

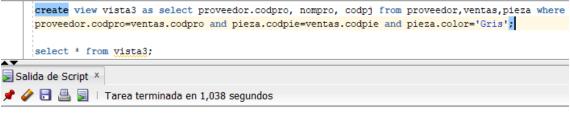
Tarea terminada en 1,09 segundos

View VISTA2 creado.

Error que empieza en la línea: 3 del comando :
insert into vista2 values('Antonio Lopez', 'Madrid')
Informe de error -
ORA-01400: no se puede realizar una inserción NULL en ("X0108849"."PROVEEDOR"."CODPRO")
```

No se puede hacer una inserción sin la clave primaria del proveedor.

4.3) Crear una vista donde aparezcan el código del proveedor, el nombre de proveedor y el código del proyecto tales que la pieza suministrada sea gris. Sobre esta vista realiza alguna consulta y enumera todos los motivos por los que sería imposible realizar una inserción.



View VISTA3 creado.

COD	NOMPRO	COD
S1	Jose Fernandez	J1
S1	Jose Fernandez	J3
S1	Jose Fernandez	J4
S1	Jose Fernandez	J1
S1	Jose Fernandez	J3
S1	Jose Fernandez	J2
S3	Luisa Gomez	J4
S3	Luisa Gomez	J1
S4	Pedro Sanchez	J3
S5	Maria Reyes	J4

10 filas seleccionadas.

Sería imposible realizar una inserción debido a que la vista no está construida sobre una única tabla y no contiene todas las llaves primarias de las tablas que la conforman.

<u>Actividad 3:</u> Realizar los ejercicios correspondientes a índices BITMAP e IOT que aparecen descritos en el cuaderno de prácticas:

- Índices BITMAP:

Cuando los valores de la clave tienen una baja cardinalidad, el uso de este tipo de índice puede ser apropiado. Se usa la cláusula BITMAP para indicar que son de este tipo de índice. Ejemplo de uso frente a índice normal:

```
    Crear tabla ejemplo: CREATE TABLE Prueba_Bit (color Varchar2(10));
```

```
2. Insertar 50000 tuplas con valores de color aleatorios. Ejecutar:
```

```
FOR i IN 1..50000 LOOP

INSERT INTO Prueba_bit (
select decode(round(dbms_random.value(1,4)),1,'Rojo',2,'Verde',
3,'Amarillo',4,'Azul') from dual);
END LOOP;
END;
```

3. Crear un índice normal: CREATE INDEX Prueba_IDX ON Prueba_Bit(color);

. Ejecutar

SELECT count(*) FROM Prueba_Bit

WHERE color='Amarillo' OR color='Azul';

- Apuntar el tiempo empleado.
- Pinchar el botón "Explicación del Plan" (F10) para ver el plan seleccionado para ejecutar esa consulta.
- Borrar indice normal: DROP INDEX Prueba_IDX;
- 8. Crear indice BITMAP:

CREATE BITMAP INDEX Prueba_BITMAP_IDX ON Prueba_Bit(color);

- 9. Volver a ejecutar la consulta anterior.
- Apuntar el tiempo empleado y comparar con el anterior.
- Pinchar el botón "Explicación del Plan" (F10) para ver el plan seleccionado para ejecutar esa consulta. Comparar con el plan usado por la consulta con el índice normal.
- Borrar tabla e indice: DROP TABLE Prueba_bit;

```
Hoja de Trabajo Generador de Consultas
      CREATE TABLE Prueba Bit (color Varchar2(10));
    BEGIN
        FOR i IN 1..50000 LOOP
              INSERT INTO Prueba_Bit(
              select decode(round(dbms_random.value(1,4)),1,'Rojo',2,'Verde',3,'Amarillo',4,'Azul') from dual);
      END:
      CREATE INDEX Prueba IDX ON Prueba Bit(color);
      SELECT count(*) FROM Prueba_Bit WHERE color='Amarillo' OR color='Azul';
Salida de Script x ► Resultado de la Consulta x
📌 🖺 🙀 🔯 SQL | Todas las Filas Recuperadas: 1 en 0,136 segundos

    COUNT(*)
            74936
Salida de Script × I Resultado de la Consulta × SExplicación del Plan ×
₱ SQL | 0,608 segundos
OPERATION
                               OBJECT_NAME
                                                     OPTIONS
                                                                          CARDINALITY
                                                                                                COST

■ SELECT STATEMENT

  AGGREGATE
    □ INLIST ITERATOR
      □ • INDEX
                               PRUEBA IDX
                                                     RANGE SCAN
                                                                                            80043
        Ė-V OR
                COLOR='Amarillo'
                COLOR='Azul'
     Other XML
       {info}
        info type="db_version"
       ···· info type="parse_schema"
      info type="dynamic_sampling" note="y"
      info type="plan_hash_full"
3912808686
       info type="plan_hash"
4138648906
      info type="plan_hash_2"
            3912808686
          {hint}
         ■ INDEX(@"SEL$1" "PRUEBA_BIT"@"SEL$1" ("PRUEBA_BIT"."COLOR"))
            OUTLINE_LEAF(@"SEL$1")
            ALL ROWS
            DB_VERSION('12.1.0.2')
            OPTIMIZER FEATURES ENABLE('12.1.0.2')
            IGNORE_OPTIM_EMBEDDED_HINTS
Hoja de Trabajo
                       Generador de Consultas
        DROP INDEX Prueba IDX;
        CREATE BITMAP INDEX Prueba_BITMAP_IDX ON Prueba_Bit(color);
        SELECT count(*) FROM Prueba Bit WHERE color='Amarillo' OR color='Azul';
屋 Salida de Script 🗴 ⊳ Resultado de la Consulta 🗴
 📌 🖺 🙀 攻 SQL 🗆 Todas las Filas Recuperadas: 1 en 0,168 segundos
         $ COUNT(*)
      1
                 74936
```

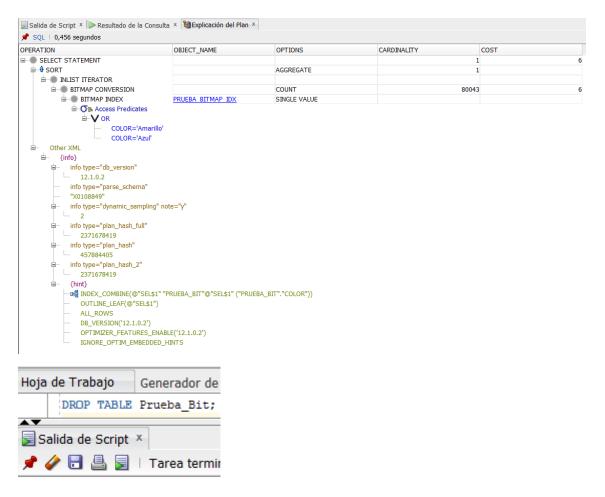
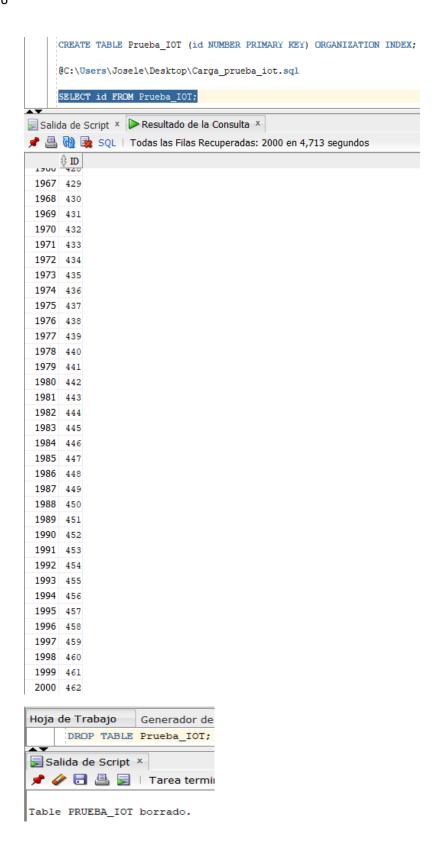


Table PRUEBA BIT borrado.

Índices IOT:

Este tipo de tablas organizan sus tuplas según el valor de la clave utilizando una estructura de árbol B*. La diferencia respecto a un índice normal estriba en que en las hojas están las tuplas, no los RID que apuntan a las tuplas. Para crear estas tablas se usa la cláusula ORGANIZATION INDEX en la sentencia CREATE TABLE, normalmente utilizan los campos de la clave primaria para ordenar la tabla. Una recuperación total devuelve los resultados ordenados por la clave. El siguiente ejemplo ilustra su uso:

- Crear la tabla: CREATE TABLE Prueba_IOT (id NUMBER PRIMARY KEY) ORGANIZATION INDEX;
- Carga de 2000 tuplas con id entre 1 y 2000, ordenadas de forma aleatoria. Abrir y ejecutar el script SQL: Carga_prueba_iot.sql.
- Ejecutar la consulta: SELECT id FROM Prueba_IOT. Observar el orden en que se obtienen las tuplas en relación al campo id.
- Pinchar el botón "Explicación del Plan" (F10) para ver el plan seleccionado para ejecutar esa consulta.
- Borrar tabla: DROP TABLE Prueba_IOT;



<u>Actividad 4:</u> En la relación de problemas EjerciciosdeÁlgebraRelacional resolver en Álgebra relacional y SQL el ejercicio 5:

```
Represendante (ANZ, nombre, direc, possinia)
(5)
         Zona- Rep ( ANI, cod-zona, polisubi, provincia)
         Pedidos (ANE, cod-art, centidad, población)
         Artiwle (col art, nombre, color, prov. Jab)
 a) Listur las provincias que son visitadas por todos las representes.
  DR AR :
    That, provincia (Sora-Rep) - That (Representante)
 * SQL.
     select provincia from Bone Rep where not exists (
              suleit and from Representante
               sulet and from Zona-Rep Zona where Zona-Rep. provincia = Zona. provincia);
      salet provincia from Bona Rep where not exists (
            solect and from Representante where not exists (
                 subst * from Some Rep Bona where Bona. ANI = Representante. ANI and Bona-Rep. province = Bona. previo)
 b) Encontrar les representantes que venden forma de sa provincia articulas fabricadas en su provincia.
 KAR:
Thepreun hunte . * ( Thepreun bunte . ANI = Sonor Rep. DNI) A (Sonor Rep. DNI = Pedidos. ANI) A
                      (Representante. provincia!= Zera-Rep. provinut) A (Representante. provinuta = Article. provinuta ) 1
(pulsos, odast=Atriba, odost) ( Representante × Zona, Rep × Pedidos × Articulos))
    select Representante. * from Representante, zona-Rep. Pedida, Article where Aspresentante. ANI = Zona-Rep. DNI and
       Representante. ANI = Pedido, DNI and Pedido, cod-art = Articlo, cod-art and Representante, provincial > Zona-Rep. provincial and
       Representante. province = Article. prov-fal ;
```

c) Obtain les pobleciones de Granable que happan reprende les 50000 euros de factura esta y quien reulisé el pedible.

TI ANI, Pedido pollarión (T(provincia = 'Cronoda' A candidad >= 50000) (Ena-Rep W Pedidos)

* SQL:

solut ANI, Podido, polhadón from Borez-Rep., Podidos where Zona-Rep. ANI = Podido. ANI and Zona-Rep., polhadón = Podidos, polhadón and Zona-Rep., provincia = "Granach" and Podido. contidad >= 30000;

d) Mostras les zones que incluyer a una sola población.

*AR

Tix (Zona-Rep) - Tizona. * (T(Zona. NNI = Zon. NNI) n(Zona. col-zona = Zon. col-zona)

1 (Zona. poblaudh 1 = Zon. poblaudh) (Zon x Zona)

* Sal:

select * from Bora. Rep where not exists (

subst * from Bora. Pep Bon where Zona. Rep. ANI = Zon. ANI and Zona. Rep. cod. zona = Zon. ced. zona and

Zona. Rep. publication <> Zon. publication);