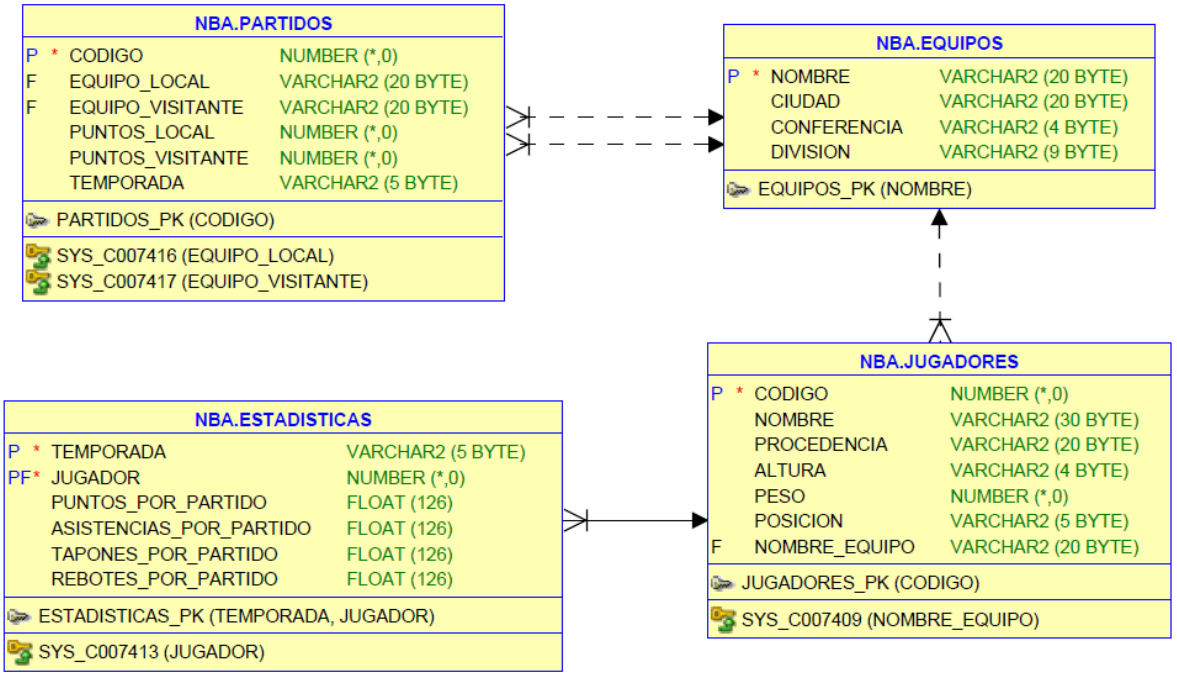
BBDD NBA

Unión tablas:

```
SELECT *
FROM PARTIDOS P, EQUIPOS E, JUGADORES J, ESTADISTICAS ES
WHERE E.NOMBRE=P.EQUIPO_LOCAL
AND J.CODIGO= ES.JUGADOR
AND J.NOMBRE_EQUIPO=E.NOMBRE;
```

```
SELECT *
FROM PARTIDOS P, EQUIPOS E, JUGADORES J, ESTADISTICAS ES
WHERE E.NOMBRE=P.EQUIPO_VISITANTE
AND J.CODIGO= ES.JUGADOR
AND J.NOMBRE_EQUIPO=E.NOMBRE;
```



CONSULTAS SQL

--1. Mostrar el nombre de todos los jugadores ordenados alfabéticamente.

```
SELECT NOMBRE
FROM JUGADORES
ORDER BY NOMBRE;
```

--2. Mostrar el nombre de los jugadores que sean pivots con mas de 200 libras

```
SELECT NOMBRE
FROM JUGADORES
WHERE UPPER(POSICION) LIKE '%C%'
AND PESO>200
ORDER BY NOMBRE;
```

--3. Mostrar el nombre de todos los equipos ordenados alfabéticamente.

```
SELECT NOMBRE
FROM EQUIPOS
ORDER BY NOMBRE;
```

--4. Mostrar el nombre de los equipos del este.

```
SELECT NOMBRE
FROM EQUIPOS
WHERE UPPER(CONFERENCIA)='EAST'
ORDER BY NOMBRE;
```

--5. Mostrar los equipos donde su ciudad empieza por c.

```
SELECT NOMBRE
FROM EQUIPOS
WHERE UPPER(CIUDAD) LIKE 'C%'
ORDER BY NOMBRE;
```

--6. Mostrar todos los jugadores y su equipo ordenado por nombre del equipo.

```
SELECT J.NOMBRE AS JUGADOR, E.NOMBRE AS EQUIPO
FROM JUGADORES J, EQUIPOS E
WHERE J.NOMBRE_EQUIPO=E.NOMBRE
ORDER BY E.NOMBRE, J.NOMBRE;
```

--7. Mostrar todos los jugadores del equipo «Raptors».

```
SELECT J.NOMBRE AS JUGADOR
FROM JUGADORES J, EQUIPOS E
WHERE J.NOMBRE_EQUIPO=E.NOMBRE
AND UPPER(E.NOMBRE)='RAPTORS'
ORDER BY J.NOMBRE;
```

--8. Mostrar los puntos por partido de ‘Pau Gasol’.

```
SELECT ES.PUNTOS_POR_PARTIDO
FROM ESTADISTICAS ES, JUGADORES J
WHERE J.CODIGO= ES.JUGADOR
AND UPPER(J.NOMBRE)='PAU GASOL';
```

--9. Mostrar los puntos por partido de ‘Pau Gasol’ en la temporada ’04/05′.

```
SELECT ES.PUNTOS_POR_PARTIDO
FROM ESTADISTICAS ES, JUGADORES J
WHERE J.CODIGO= ES.JUGADOR
AND UPPER(J.NOMBRE)='PAU GASOL'
AND TO_CHAR(ES.TEMPORADA)= '04/05';
```

--10. Mostrar el numero de puntos de cada jugador en toda su carrera.

```
SELECT J.NOMBRE, SUM(ES.PUNTOS_POR_PARTIDO) AS PUNTOS_CARRERA
FROM ESTADISTICAS ES, JUGADORES J
WHERE J.CODIGO= ES.JUGADOR
GROUP BY J.NOMBRE;
```

--11. Mostrar el número de jugadores de cada equipo.

```
SELECT E.NOMBRE AS EQUIPO, COUNT(J.NOMBRE) AS TOTAL_JUGADORES
FROM JUGADORES J, EQUIPOS E
WHERE J.NOMBRE_EQUIPO=E.NOMBRE
GROUP BY E.NOMBRE
ORDER BY E.NOMBRE;
```

--12. Mostrar el jugador que mas puntos ha realizado en toda su carrera.

```
SELECT J.NOMBRE, SUM(ES.PUNTOS_POR_PARTIDO) AS PUNTOS_CARRERA
FROM ESTADISTICAS ES, JUGADORES J
WHERE J.CODIGO= ES.JUGADOR
GROUP BY J.NOMBRE
HAVING SUM(ES.PUNTOS_POR_PARTIDO)= (SELECT MAX(SUM(ES.PUNTOS_POR_PARTIDO))
                        FROM ESTADISTICAS ES, JUGADORES J
                        WHERE J.CODIGO= ES.JUGADOR
                        GROUP BY J.NOMBRE);
```

--13. Mostrar el nombre del equipo, conferencia y división del jugador más alto de la NBA.

```
SELECT J.NOMBRE AS JUGADOR, E.NOMBRE AS EQUIPO, E.CONFERENCIA,
E.DIVISION, MAX(J.ALTURA)
FROM JUGADORES J, EQUIPOS E
WHERE J.NOMBRE_EQUIPO=E.NOMBRE
GROUP BY J.NOMBRE, E.NOMBRE, E.CONFERENCIA, E.DIVISION
HAVING MAX(J.ALTURA)= (SELECT MAX(MAX(J.ALTURA))
                FROM JUGADORES J, EQUIPOS E
                WHERE J.NOMBRE_EQUIPO=E.NOMBRE
                GROUP BY E.NOMBRE);
```

--14. Mostrar la suma de los puntos por partido de todos los jugadores españoles donde el equipo donde juegan este en ‘Los Ángeles’.

```
SELECT SUM(ES.PUNTOS_POR_PARTIDO) AS PUNTOS
FROM ESTADISTICAS ES, JUGADORES J, EQUIPOS E
WHERE J.CODIGO= ES.JUGADOR
AND J.NOMBRE_EQUIPO=E.NOMBRE
AND UPPER(J.PROCEDENCIA)='SPAIN'
AND UPPER(E.CIUDAD)='LOS ANGELES';
```

--15. Mostrar la media de puntos en partidos de los equipos de la división Pacific.

```
SELECT TRUNC(AVG(NVL(ES.PUNTOS_POR_PARTIDO, 0))) AS
MEDIA_PUNTOS_POR_PARTIDO
FROM ESTADISTICAS ES, JUGADORES J, EQUIPOS E
WHERE J.CODIGO= ES.JUGADOR
AND J.NOMBRE_EQUIPO=E.NOMBRE
AND UPPER(E.DIVISION)='PACIFIC';
```

```
select avg(puntos)
```

```
from (select sum(puntos_local) as puntos
    from PARTIDOS
    where equipo_local in (select nombre
                    from equipos
                    where lower(DIVISION) = 'pacific')
    union
    select sum(puntos_visitante) as puntos
    from PARTIDOS
    where equipo_visitante in (select nombre
                    from equipos
                    where lower(DIVISION) = 'pacific')) t;
SELECT AVG(SUM(PUNTOS))
FROM(
        SELECT E.NOMBRE, (P.PUNTOS_LOCAL) AS PUNTOS
        FROM PARTIDOS P, EQUIPOS E
        WHERE E.NOMBRE=P.EQUIPO_LOCAL
        AND UPPER(E.DIVISION)='PACIFIC'
        UNION ALL
        SELECT E.NOMBRE,(P.PUNTOS_VISITANTE)
        FROM PARTIDOS P, EQUIPOS E
        WHERE E.NOMBRE=P.EQUIPO_VISITANTE
        AND UPPER(E.DIVISION)='PACIFIC')
GROUP BY NOMBRE;
```

--16. Mostrar el partido o partidos (equipo_local, equipo_visitante y diferencia) con mayor diferencia de puntos.

```
SELECT DISTINCT P.EQUIPO_LOCAL,P.EQUIPO_VISITANTE
,ABS(SUM(P.PUNTOS_LOCAL)-SUM(P.PUNTOS_VISITANTE)) AS DIFERENCIA
FROM PARTIDOS P
GROUP BY P.EQUIPO_LOCAL,P.EQUIPO_VISITANTE
HAVING ABS(SUM(P.PUNTOS_LOCAL)-SUM(P.PUNTOS_VISITANTE))= (SELECT
DISTINCT MAX(ABS(SUM(P.PUNTOS_LOCAL)-SUM(P.PUNTOS_VISITANTE)))
                                        FROM PARTIDOS P
                                        GROUP BY
P.EQUIPO_LOCAL,P.EQUIPO_VISITANTE);
```

*→la real solución es la de abajo, lo anterior es la diferencia mayor de suma de puntos por partidos entre equipos**********/*

```
SELECT DISTINCT P.EQUIPO_LOCAL,P.EQUIPO_VISITANTE,
ABS(P.PUNTOS_LOCAL-P.PUNTOS_VISITANTE)
FROM PARTIDOS P
WHERE ABS(P.PUNTOS_LOCAL-P.PUNTOS_VISITANTE) = (SELECT
MAX(ABS(P.PUNTOS_LOCAL - P.PUNTOS_VISITANTE))
                                FROM PARTIDOS P);
```

--18. Mostrar los puntos de cada equipo en los partidos, tanto de local como de visitante. Usa una vista

```
CREATE VIEW V_PUNTOS_EQUIPOS(EQUIPO, TOTAL_PUNTOS)
 AS
SELECT J.NOMBRE_EQUIPO, SUM(ES.PUNTOS_POR_PARTIDO)
FROM JUGADORES J, ESTADISTICAS ES
WHERE J.CODIGO= ES.JUGADOR
GROUP BY  J.NOMBRE_EQUIPO;

SELECT *
FROM V_PUNTOS_EQUIPOS;
```

--19. Mostrar quien gana en cada partido (codigo, equipo_local, equipo_visitante, equipo_ganador), en caso de empate será null.

```
SELECT DISTINCT P.CODIGO, p.equipo_local, p.equipo_visitante,
       CASE
          WHEN P.PUNTOS_LOCAL > P.PUNTOS_VISITANTE THEN P.EQUIPO_LOCAL
          WHEN P.PUNTOS_LOCAL < P.PUNTOS_VISITANTE THEN P.EQUIPO_VISITANTE
          ELSE null
        END AS EQUIPO_GANADOR
FROM PARTIDOS P;
```

--MOSTRAR la información de los equipos que tienen más cantidad de puntos anotados como local o visitante

```
SELECT P.EQUIPO_LOCAL AS NOMBRE_EQUIPO, SUM(P.PUNTOS_LOCAL) AS
TOTAL_PUNTOS
FROM PARTIDOS P, EQUIPOS E
WHERE E.NOMBRE=P.EQUIPO_LOCAL
GROUP BY P.EQUIPO_LOCAL
HAVING SUM(P.PUNTOS_LOCAL)=(SELECT MAX(SUM(P.PUNTOS_LOCAL))
                    FROM PARTIDOS P, EQUIPOS E
                    WHERE E.NOMBRE=P.EQUIPO_LOCAL
                    GROUP BY P.EQUIPO_LOCAL)
UNION ALL

SELECT P.EQUIPO_VISITANTE, SUM(P.PUNTOS_VISITANTE)
FROM PARTIDOS P, EQUIPOS E
WHERE E.NOMBRE=P.EQUIPO_VISITANTE
GROUP BY P.EQUIPO_VISITANTE
HAVING SUM(P.PUNTOS_VISITANTE)=(SELECT MAX(SUM(P.PUNTOS_VISITANTE))
                    FROM PARTIDOS P, EQUIPOS E
                    WHERE E.NOMBRE=P.EQUIPO_VISITANTE
                    GROUP BY P.EQUIPO_VISITANTE);
```

PLSQL

/*Haz un procedimiento que, dada una temporada y un equipo (ambos pasados como parámetros de entrada) diga:
Cuántos partidos en total ha ganado ese equipo como local

Para cada partido: Decir cuántos puntos anotó.*/

```
SELECT P.EQUIPO_LOCAL, P.PUNTOS_LOCAL, P.PUNTOS_VISITANTE,
P.EQUIPO_VISITANTE
        FROM PARTIDOS P
        WHERE P.TEMPORADA='98/99'
        AND UPPER(P.EQUIPO_LOCAL)= 'MAGIC';
SET SERVEROUTPUT ON
/
CREATE OR REPLACE PROCEDURE spPuntosPartido(P_TEMPORADA
PARTIDOS.TEMPORADA%TYPE, P_EQUIPO EQUIPOS.NOMBRE%TYPE)
IS
     CURSOR C_PARTIDOS_GANADOS IS
        SELECT P.EQUIPO_LOCAL, P.PUNTOS_LOCAL, P.PUNTOS_VISITANTE,
P.EQUIPO_VISITANTE
        FROM PARTIDOS P
        WHERE P.TEMPORADA=P_TEMPORADA
        AND UPPER(P.EQUIPO_LOCAL)= P_EQUIPO;

        V_LOCAL PARTIDOS.EQUIPO_LOCAL%TYPE;
        V_PTOS_LOCAL PARTIDOS.PUNTOS_LOCAL%TYPE;
        V_PTOS_VISITA PARTIDOS.PUNTOS_VISITANTE%TYPE;
        V_VISITA PARTIDOS.EQUIPO_VISITANTE%TYPE;
        V_CONTADOR_LOCAL NUMBER:=0;

BEGIN
   OPEN C_PARTIDOS_GANADOS;
        LOOP
           FETCH C_PARTIDOS_GANADOS INTO V_LOCAL, V_PTOS_LOCAL,
V_PTOS_VISITA,V_VISITA;
           EXIT WHEN C_PARTIDOS_GANADOS%NOTFOUND;
           DBMS_OUTPUT.PUT_LINE(V_LOCAL || ' HA ANOTADO ' || V_PTOS_LOCAL || '
PUNTOS CONTRA ' || V_VISITA);
           IF V_PTOS_LOCAL> V_PTOS_VISITA THEN
              V_CONTADOR_LOCAL:=V_CONTADOR_LOCAL+1;

           END IF;

        END LOOP;

  CLOSE C_PARTIDOS_GANADOS;
  DBMS_OUTPUT.PUT_LINE(V_LOCAL || ' HA GANADO ' || V_CONTADOR_LOCAL || '
PARTIDOS');

  EXCEPTION
  WHEN OTHERS THEN
     DBMS_OUTPUT.PUT_LINE('Ocurrió el error ' ||SQLCODE ||' ---mensaje: ' ||
SQLERRM);
```

```
END;
/
BEGIN
   spPuntosPartido('98/99', 'MAGIC');
END;
/
--otra manera de hacerlo, usando un cursor explícito y otro implícito:
CREATE OR REPLACE PROCEDURE spEquipoGanador(pTemp partidos.temporada%type,
pEquipo equipos.nombre%type)
is
   v_partidos_ganados number;

   cursor c_part is
      select pa.codigo, pa.puntos_local
      from partidos pa
      where upper(pa.equipo_local) = upper(pEquipo)
      and pa.temporada = pTemp;

   v_cod partidos.codigo%type;
   v_puntos_local partidos.puntos_local%type;

begin

   open c_part;
   loop
      fetch c_part into v_cod, v_puntos_local;
      exit when c_part%notfound;

      DBMS_OUTPUT.PUT_LINE('El equipo '|| pEquipo || ' en el partido '||v_cod||' metio '||
v_puntos_local);

   end loop;
   close c_part;

   --- y cuantos partidos ha ganado
   select count(*) into v_partidos_ganados
   from partidos pa
   where pa.puntos_visitante < pa.puntos_local
   and upper(pa.equipo_local) = upper(pEquipo)
   and pa.temporada = pTemp;

   DBMS_OUTPUT.PUT_LINE('El equipo '|| pEquipo || ' ganó '||v_partidos_ganados||'
partidos ');

EXCEPTION
   WHEN OTHERS THEN
   DBMS_OUTPUT.PUT_LINE('Ocurrió el error ' ||SQLCODE ||' ---mensaje: ' || SQLERRM);
end;
```

/* REALIZAR EN LA BASE DE DATOS LAS MODIFICACIONES NECESARIAS PARA QUE LOS BULLS DE CHICAGO NO PUEDAN PERDER UN PARTIDO **/**

```
CREATE OR REPLACE TRIGGER T_BullsSiempreGanan
BEFORE
INSERT OR UPDATE OF EQUIPO_LOCAL OR UPDATE OF EQUIPO_VISITANTE OR
UPDATE OF PUNTOS_LOCAL OR UPDATE OF PUNTOS_VISITANTE
ON PARTIDOS
FOR EACH ROW
DECLARE

BEGIN
      IF :NEW.PUNTOS_LOCAL< :NEW.PUNTOS_VISITANTE AND
UPPER(:NEW.EQUIPO_LOCAL) = 'BULLS' THEN
         raise_application_error(-20600,'LOS BULLS NO PUEDEN TENER MENOS
PUNTOS QUE EL EQUIPO CONTRARIO');
      END IF;

      IF :NEW.PUNTOS_LOCAL> :NEW.PUNTOS_VISITANTE AND
UPPER(:NEW.EQUIPO_VISITANTE) = 'BULLS' THEN
         raise_application_error(-20600,'LOS BULLS NO PUEDEN TENER MENOS
PUNTOS QUE EL EQUIPO CONTRARIO');
      END IF;

END;
/
```

/* CREAR UNA FUNCIÓN QUE DADA UNA DIVISIÓN Y UNA TEMPORADA DEVUELVA LOS 3 EQUIPOS (NOMBRES CONCATENADOS) QUE MÁS PUNTOS HAN METIDO EN ESA DIVISIÓN Y TEMPORADA*/**

```
SELECT *
FROM(
         SELECT NOMBRE, SUM(TOTAL_PUNTOS)
         FROM(
               SELECT E.NOMBRE, SUM(P.PUNTOS_LOCAL) AS TOTAL_PUNTOS
               FROM PARTIDOS P, EQUIPOS E
               WHERE E.NOMBRE=P.EQUIPO_LOCAL
               AND UPPER(E.DIVISION)='CENTRAL'
               AND UPPER(P.TEMPORADA)='98/99'
               GROUP BY E.NOMBRE
               UNION ALL
               SELECT E.NOMBRE, SUM(P.PUNTOS_VISITANTE)
               FROM PARTIDOS P, EQUIPOS E
               WHERE E.NOMBRE=P.EQUIPO_VISITANTE
               AND UPPER(E.DIVISION)='CENTRAL'
```

```
                AND UPPER(P.TEMPORADA)='98/99'
                GROUP BY E.NOMBRE)
        GROUP BY NOMBRE
        ORDER BY 2 DESC)
WHERE ROWNUM <=3;
/
CREATE OR REPLACE FUNCTION fLos3MejoresEQUIPOS (p_DIVISION
EQUIPOS.DIVISION%TYPE, P_TEMPORADA PARTIDOS.TEMPORADA%TYPE)
RETURN VARCHAR2
IS
v_lista varchar2(200);
  CURSOR C_MAS_PUNTOS IS

      SELECT NOMBRE, SUM(TOTAL_PUNTOS)
      FROM(
            SELECT E.NOMBRE, SUM(P.PUNTOS_LOCAL) AS TOTAL_PUNTOS
            FROM PARTIDOS P, EQUIPOS E
            WHERE E.NOMBRE=P.EQUIPO_LOCAL
            AND UPPER(E.DIVISION)=UPPER(p_DIVISION)
            AND UPPER(P.TEMPORADA)=UPPER(P_TEMPORADA)
            GROUP BY E.NOMBRE
            UNION ALL
            SELECT E.NOMBRE, SUM(P.PUNTOS_VISITANTE)
            FROM PARTIDOS P, EQUIPOS E
            WHERE E.NOMBRE=P.EQUIPO_VISITANTE
            AND UPPER(E.DIVISION)=UPPER(p_DIVISION)
            AND UPPER(P.TEMPORADA)=UPPER(P_TEMPORADA)
            GROUP BY E.NOMBRE)
      GROUP BY NOMBRE
      ORDER BY 2 DESC;

        V_EQUIPO EQUIPOS.NOMBRE%TYPE;
        V_TOTAL_PUNTOS NUMBER;
BEGIN
    OPEN C_MAS_PUNTOS;
        for x in 1..3 LOOP
           FETCH C_MAS_PUNTOS INTO V_EQUIPO, V_TOTAL_PUNTOS;
           EXIT WHEN C_MAS_PUNTOS%NOTFOUND;
           v_lista:=v_lista||'--> '||v_equipo;
           DBMS_OUTPUT.PUT_LINE(V_equipo || ' HA OBTENIDO ' ||
V_TOTAL_PUNTOS|| ' PUNTOS');
        END LOOP;

   CLOSE C_MAS_PUNTOS;

   RETURN v_lista;

   EXCEPTION
```

```
   WHEN OTHERS THEN
      DBMS_OUTPUT.PUT_LINE('Ocurrió el error ' ||SQLCODE ||' ---mensaje: ' ||
SQLERRM);
        return SQLERRM;
END;

/
begin

   dbms_output.put_line('LOS MEJORES EQUIPOS SON ' ||
fLos3MejoresEQUIPOS('CENTRAL', '98/99'));
end;
/
```