```
create table Aule (
AID integer,
ANome varchar2(20),
AIndirizzo varchar2(50),
ACapienza integer,
primary key (AID));

create table Docenti (
DID integer,
DNomeCognome varchar2(50),
primary key (DID));

create table Prenotazioni (
PAulaID integer,
PDocenteID integer,
PData date,
POraDa integer
POraA integer
primary key (PAulaID,PDocenteID,PDATA,POraDa),
foreign key (PAulaID) references Aule(AID),
foreign key (PDocenteID) references Docenti(DID));

 CREATE OR REPLACE PROCEDURE PRENOTA(vDocenteId DOCENTI.DID%TYPE, vData date,
                         vOraDa integer, vOraA integer, vCapienzaMin integer) IS
  CURSOR cAuleLibere(vData1 Date) IS
    SELECT AID
    FROM Aule
    WHERE ACapienza >vCapienzaMin AND
          AID not in (SELECT PAulaID from Prenotazioni
                      WHERE PData =vData1 AND (vOraDa>POraA OR vOraA < POraDa))
    ORDER BY ACapienza ASC;

   vAulaLibera integer;
   vFound BOOLEAN := FALSE;

 BEGIN

   OPEN cAuleLibere(vData);
     LOOP
        FETCH cAuleLibere into vAulaLibera;
        EXIT WHEN cAuleLibere%NOTFOUND;
        vFound:=TRUE;
        INSERT INTO PRENOTAZIONI
        VALUES (vAulaLibera,vDocenteId, vData, vOraDa, vOraA);
    END LOOP;
    CLOSE cAuleLibere;
    IF (vFound = FALSE) THEN
     FOR i IN 1..5
     LOOP
       OPEN cAuleLibere(vData+i);
       LOOP
          FETCH cAuleLibere into vAulaLibera;
         EXIT WHEN cAuleLibere%NOTFOUND;
          DBMS_OUTPUT.PUT_LINE('Data: ' || vData+i || 'Aula: ' || vAulaLibera);
       END LOOP;
     END LOOP;
  END IF;
 END;
```

```
explain plan for
select N_NAME
from NATION,CUSTOMER,ORDERS
where N_NATIONKEY=C_NATIONKEY and C_CUSTKEY=O_CUSTKEY
group by N_NATIONKEY, N_NAME
having count(*) > 60000;
@?/RDBMS/ADMIN/UTLXPLS;
```

```
Plan Table
--------------------------------------------------------------------------------
| Operation                 |  Name    |  Rows | Bytes|  Cost  | Pstart| Pstop |
--------------------------------------------------------------------------------
| SELECT STATEMENT          |          |    23 |  966 | 119132 |       |       |
|  FILTER                   |          |       |      |        |       |       |
|   SORT GROUP BY           |          |    23 |  966 | 119132 |       |       |
|    HASH JOIN              |          |     1M|   60M|  10640 |       |       |
|     TABLE ACCESS FULL     |NATION    |    25 |  725 |      1 |       |       |
|     MERGE JOIN            |          |     1M|   18M|  10126 |       |       |
|      SORT JOIN            |          |     1M|    7M|        |       |       |
|       INDEX FAST FULL SCAN|IX_CUST_O |     1M|    7M|   6849 |       |       |
|      SORT JOIN            |          |   150K|    1M|   2218 |       |       |
|       TABLE ACCESS FULL   |CUSTOMER  |   150K|    1M|   1059 |       |       |
--------------------------------------------------------------------------------
```

$NP_N = \lceil 25 \times 105/ (4096 \times 0{,}69) \rceil = 1$
$NP_C = \lceil 150.000 \times 158/ (4096 \times 0{,}69) \rceil = 8.386$
$NL_{O\_CUSTKEY} = \lceil (150.000 \times 4 + 1.500.000 \times 4) / (4.096 \times 0{,}69) \rceil = 2.336$

L'indice IX_CUST_ORDERS indicizza 1.500.000 ordini in base ai 150.000 clienti a cui sono intestati.
ORACLE utilizza il fast full scan poiché non ha necessità di accedere a ORDERS gli basta l'attributo O_CUSTKEY che è contenuto nell'indice. ATTENZIONE il fast full scan non sostituisce l'ordinamento (vedi pag 91 del manuale Design & Tuning Performance) CUSTOMER è già ordinata sulla chiave.

Customer è già ordinata sulla chiave, ma nel piano di esecuzione è inclusa comunque l'operazione di ordinamento (entrambe le soluzioni saranno considerate corrette.).

Sort(Customer)= $2 \times 8.386 \times (\lceil \log_{100} 8.386 \rceil + 1) = 2 \times 8.386 \times 3 = 50.316$
Sort(IX_CUST_O)= $2 \times 2.336 \times (\lceil \log_{100} 2.336 \rceil + 1) = 2 \times 2.336 \times 3 = 14.016$

JOIN(C - O) = 14.016 + 50.316 + 2.336 + 8.386 = **75.054**

Il sistema non accede a ORDERS quindi la tabella risultato include solo O_ORDERKEY che è lungo 4 byte

$NP_{C+O} = \lceil 1.500.000 \times (158+4)/ (4096 \times 0{,}69) \rceil = 85.980$

JOIN(C+O - N) = $3 \times (85.980+1)$ = **257.943**

$NT_{C+O+N} = 1.500.000$
$NP_{C+O+N} = \lceil 1.500.000 \times (158+4+105) / (4.096 \times 0{,}69) \rceil = 557.278$

Costo Group By=$2 \times 557.278 \times (\lceil \log_{100} 557.278 \rceil + 1) = 2 \times 557.278 \times 4$ =**4.458.224**

$NP_{GB(N\_NAME,N\_NATIONKEY)} = \lceil 25 \times (25+4) / (4.096 \times 0,69) \rceil = 1$

Costo totale = 75.054 + 257.943 + 4.458.224 + 1= 4.791.222