INFS 3200: Practice One
Name: Peng Yu
Student ID: 46635884

Task 1

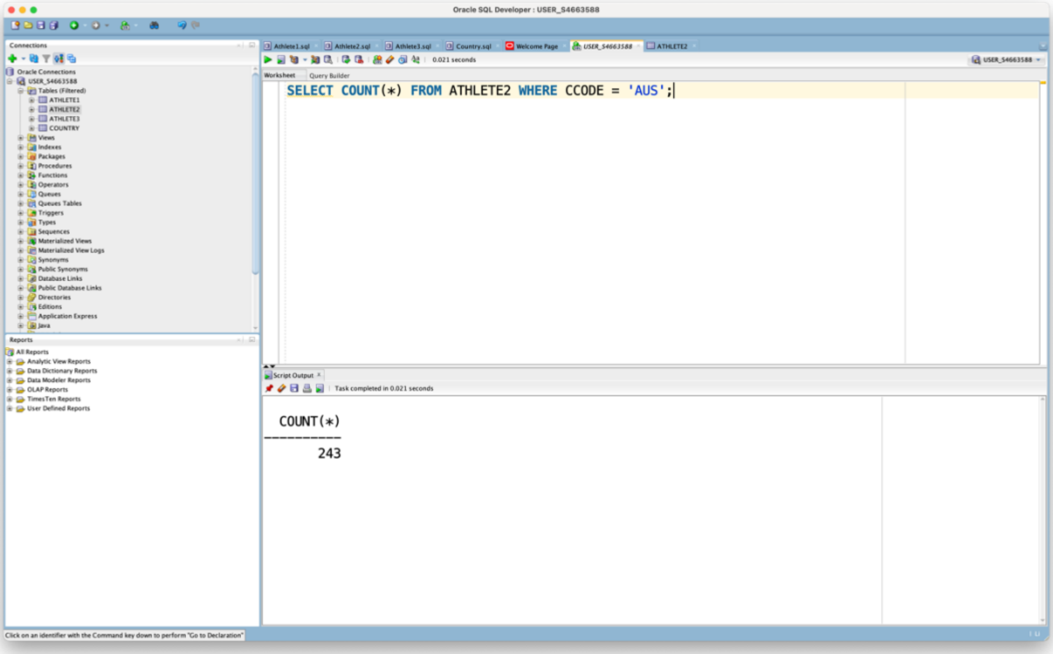
(1) Answer:

Code:

```
SELECT COUNT(*)
FROM ATHLETE2
WHERE CCODE = 'AUS';
```

Result:

COUNT(*)
243



```
SELECT COUNT(*) FROM ATHLETE2 WHERE CCODE = 'AUS';
```

Task completed in 0.021 seconds

```
COUNT(*)
----------
       243
```

(2) Answer:

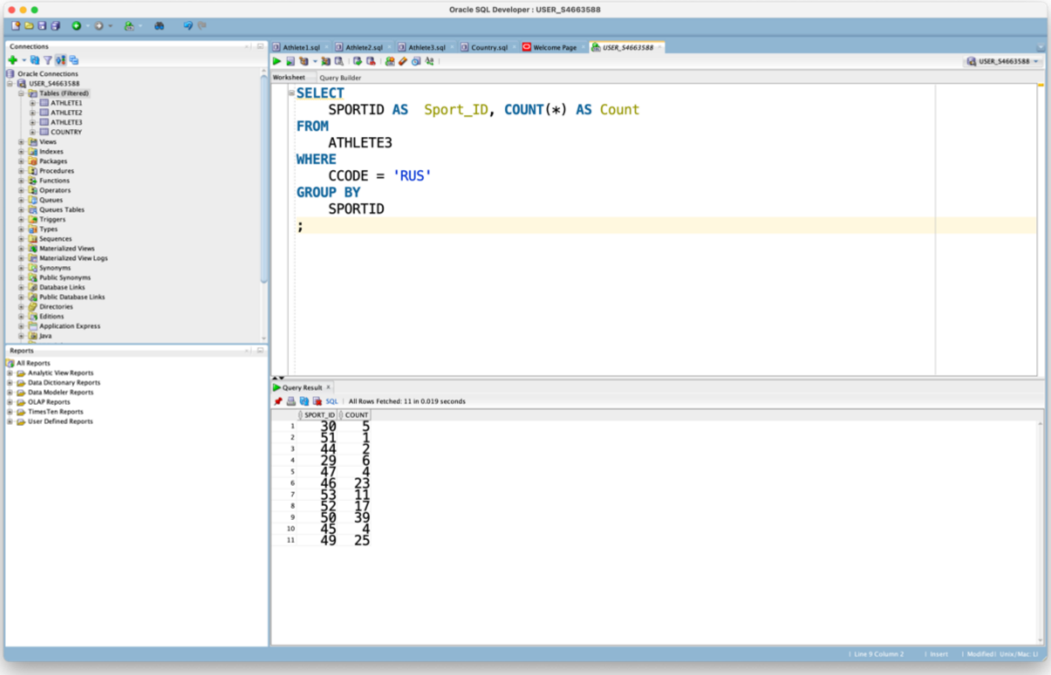
Code:

```
SELECT SPORTID AS Sport_ID, COUNT(*) AS Count
FROM ATHLETE3
```

WHERE CCODE = 'RUS'
GROUP BY SPORTID;

Result:

	Sport_ID	COUNT
1	30	5
2	51	1
3	44	2
4	29	6
5	47	4
6	46	23
7	53	11
8	52	17
9	50	39
10	45	4
11	49	25



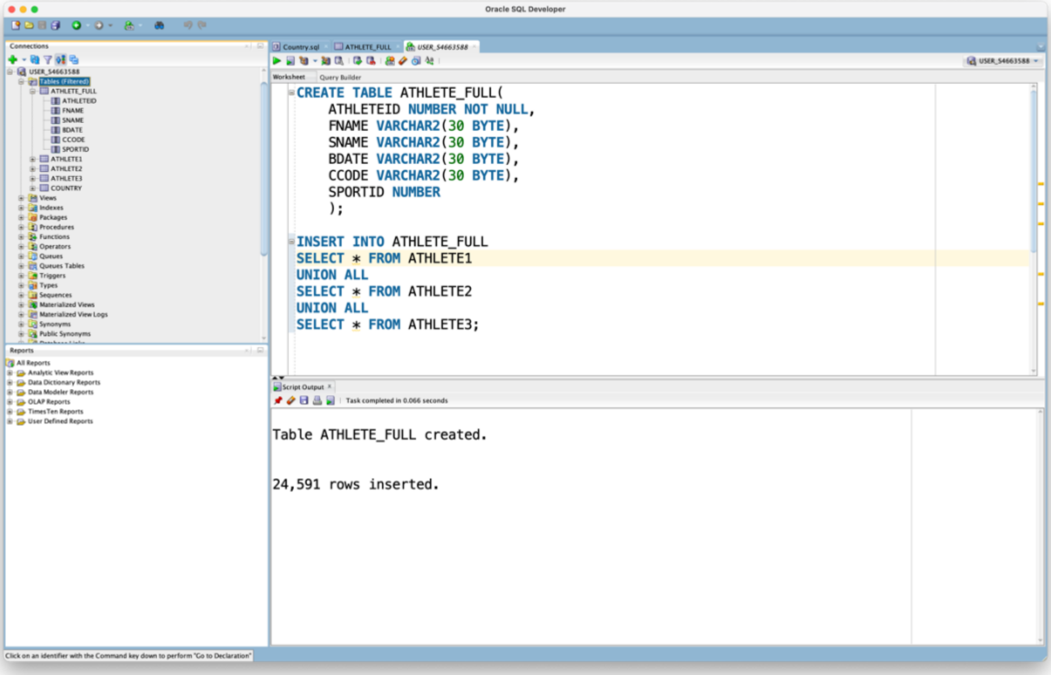
```
SELECT
    SPORTID AS  Sport_ID, COUNT(*) AS Count
FROM
    ATHLETE3
WHERE
    CCODE = 'RUS'
GROUP BY
    SPORTID
;
```

All Rows Fetched: 11 in 0.019 seconds

```
    SPORT_ID  COUNT
 1        30      5
 2        51      1
 3        44      2
 4        29      6
 5        47      4
 6        46     23
 7        53     11
 8        52     17
 9        50     39
10        45      4
11        49     25
```

(3) Answer:

First, we combine all the tables into a single table.



```
CREATE TABLE ATHLETE_FULL(
    ATHLETEID NUMBER NOT NULL,
    FNAME VARCHAR2(30 BYTE),
    SNAME VARCHAR2(30 BYTE),
    BDATE VARCHAR2(30 BYTE),
    CCODE VARCHAR2(30 BYTE),
    SPORTID NUMBER
    );

INSERT INTO ATHLETE_FULL
SELECT * FROM ATHLETE1
UNION ALL
SELECT * FROM ATHLETE2
UNION ALL
SELECT * FROM ATHLETE3;
```

```
Table ATHLETE_FULL created.


24,591 rows inserted.
```

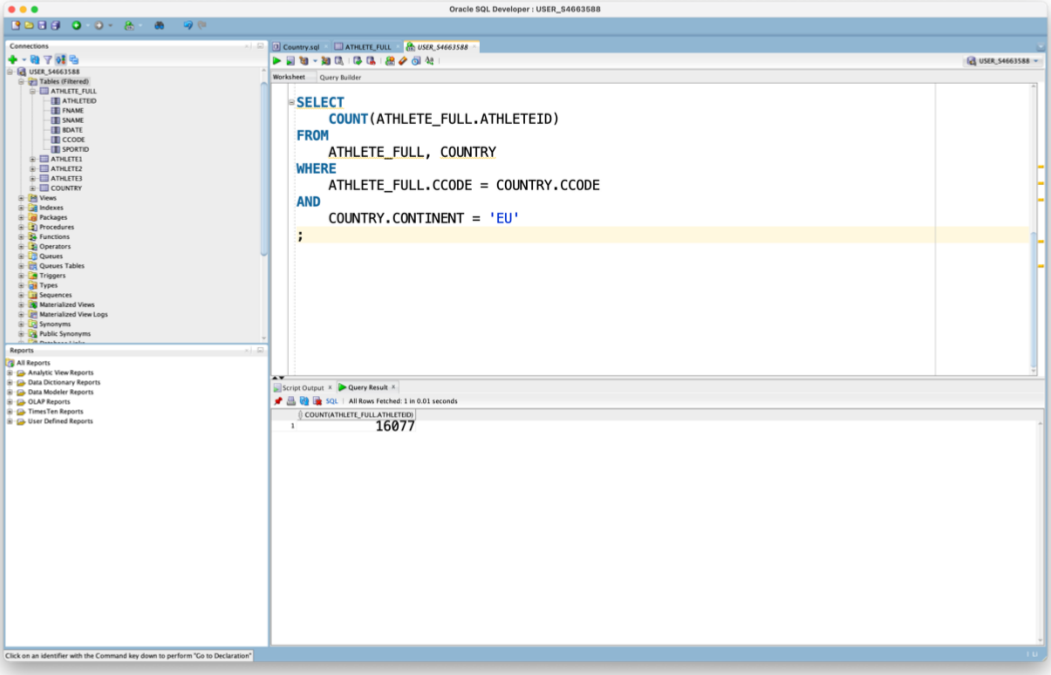
And then follow the instructions and get the results

Code:

```
CREATE TABLE ATHLETE_FULL(
	ATHLETEID NUMBER NOT NULL,
	FNAME VARCHAR2(30 BYTE),
	SNAME VARCHAR2(30 BYTE),
	BDATE VARCHAR2(30 BYTE),
	CCODE VARCHAR2(30 BYTE),
	SPORTID NUMBER
);

INSERT INTO ATHLETE_FULL
SELECT * FROM ATHLETE1
UNION ALL
SELECT * FROM ATHLETE2
UNION ALL
SELECT * FROM ATHLETE3;

SELECT COUNT(ATHLETE_FULL.ATHLETEID)
FROM ATHLETE_FULL, COUNTRY
WHERE ATHLETE_FULL.CCODE = COUNTRY.CCODE
	AND COUNTRY.CONTINENT = 'EU';
```

Result:

COUNT(ATHLETE_FULL.ATHLETEID)
16077



```sql
SELECT
    COUNT(ATHLETE_FULL.ATHLETEID)
FROM
    ATHLETE_FULL, COUNTRY
WHERE
    ATHLETE_FULL.CCODE = COUNTRY.CCODE
AND
    COUNTRY.CONTINENT = 'EU'
;
```

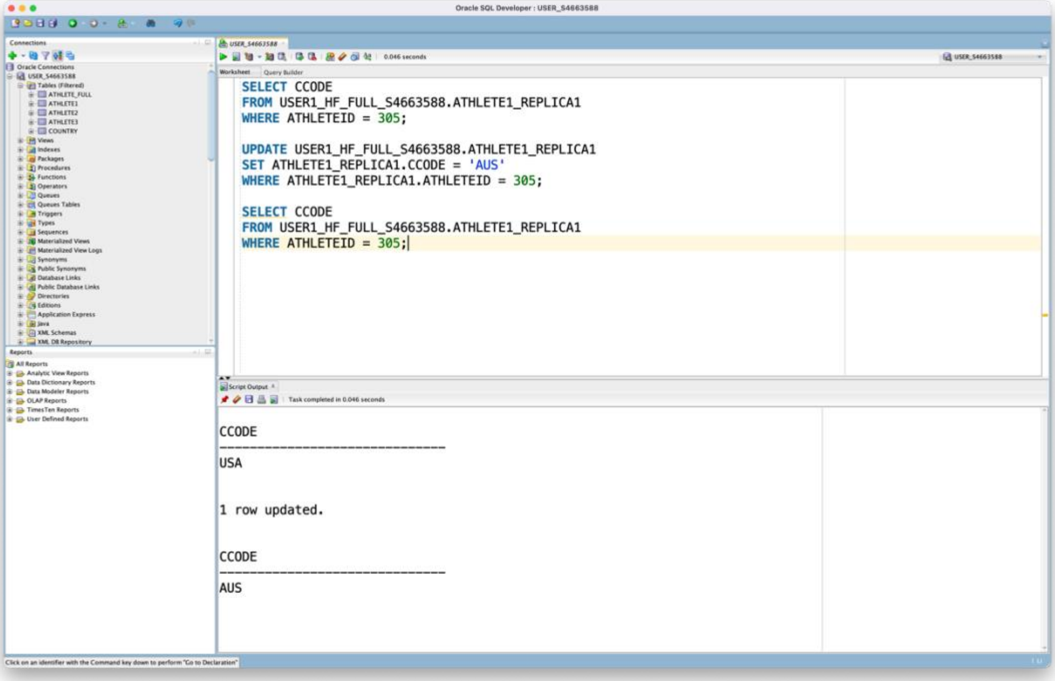
```
All Rows Fetched: 1 in 0.01 seconds

  COUNT(ATHLETE_FULL.ATHLETEID)
1                         16077
```

TASK 2

Answer:

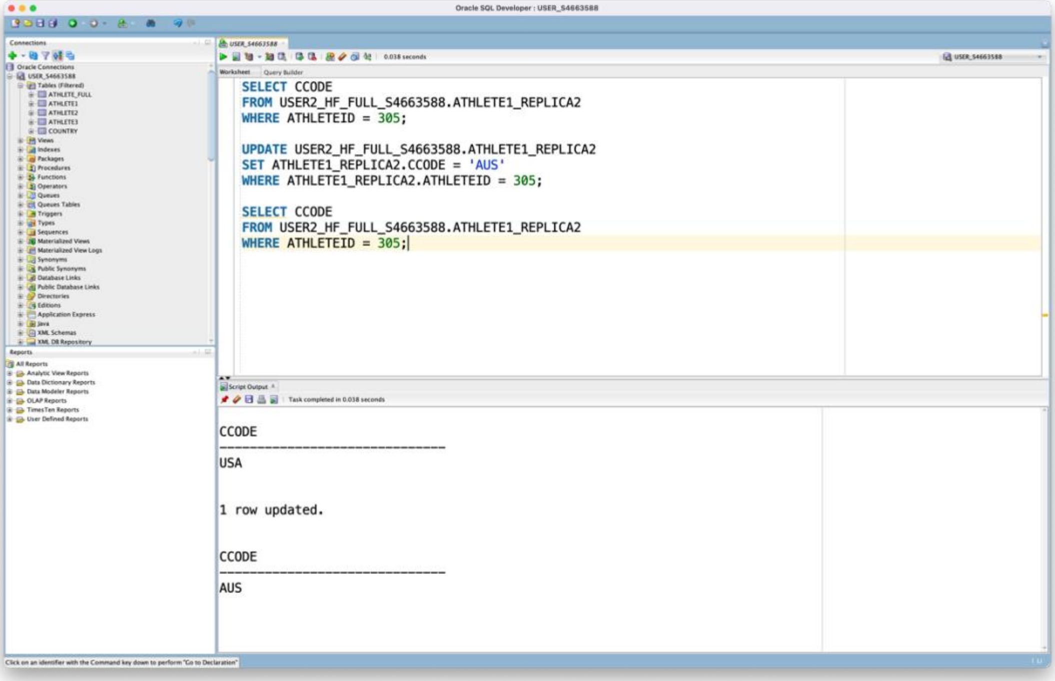
The change of ID is 305. We know that the table is Horizontal fragmentation. The condition of fragmentation is Athlete1: 1<= AthleteID < 7656 , Athlete2: 7657<= AthleteID < 17318 and Athlete3: 17319 <= AthleteID <= 24591. So we only need to change the information in table Athlete1.

Due to Full Replication, each site has a full copy of each fragment, so each site needs to be modified.



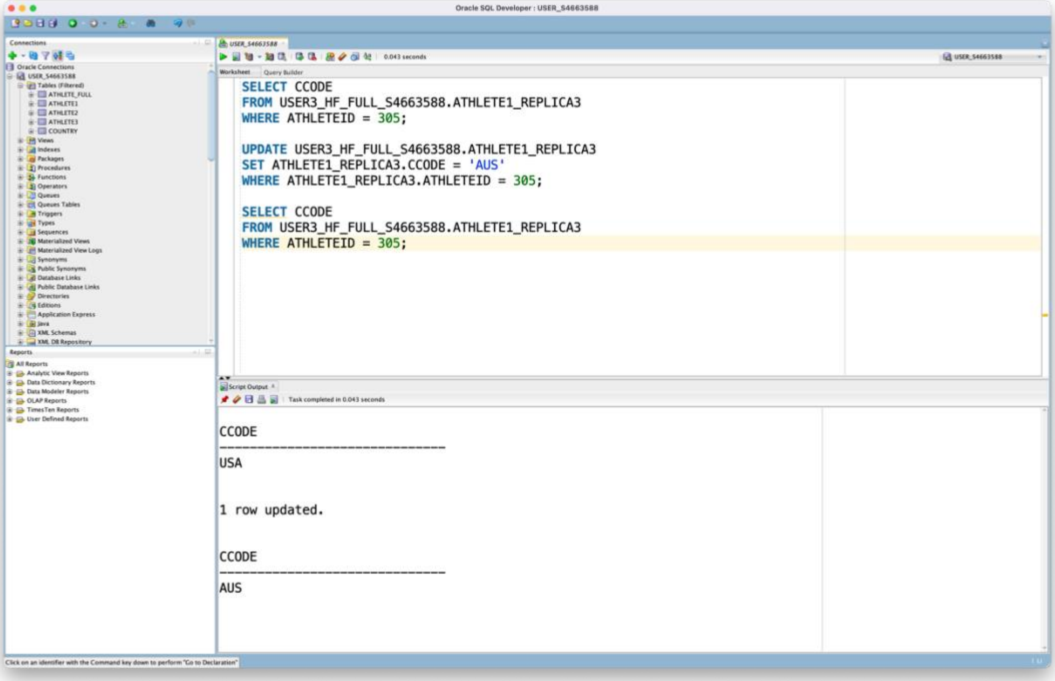
```
SELECT CCODE
FROM USER1_HF_FULL_S4663588.ATHLETE1_REPLICA1
WHERE ATHLETEID = 305;

UPDATE USER1_HF_FULL_S4663588.ATHLETE1_REPLICA1
SET ATHLETE1_REPLICA1.CCODE = 'AUS'
WHERE ATHLETE1_REPLICA1.ATHLETEID = 305;

SELECT CCODE
FROM USER1_HF_FULL_S4663588.ATHLETE1_REPLICA1
WHERE ATHLETEID = 305;
```

```
CCODE
------------------------------
USA


1 row updated.


CCODE
------------------------------
AUS
```



```
SELECT CCODE
FROM USER2_HF_FULL_S4663588.ATHLETE1_REPLICA2
WHERE ATHLETEID = 305;

UPDATE USER2_HF_FULL_S4663588.ATHLETE1_REPLICA2
SET ATHLETE1_REPLICA2.CCODE = 'AUS'
WHERE ATHLETE1_REPLICA2.ATHLETEID = 305;

SELECT CCODE
FROM USER2_HF_FULL_S4663588.ATHLETE1_REPLICA2
WHERE ATHLETEID = 305;
```

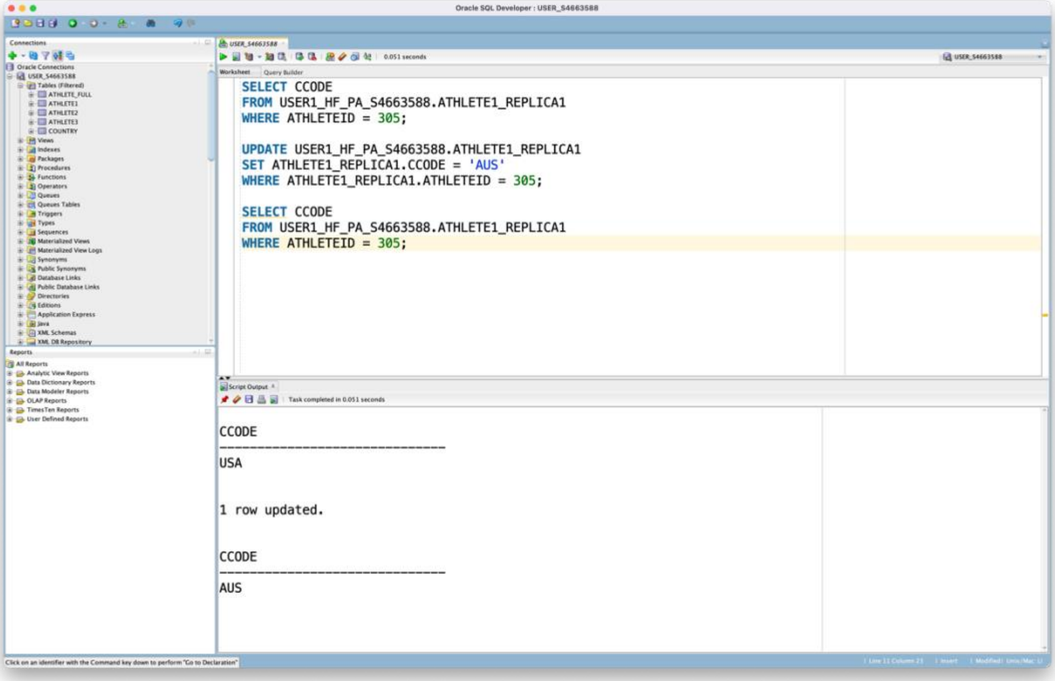
```
CCODE
------------------------------
USA


1 row updated.


CCODE
------------------------------
AUS
```



```sql
SELECT CCODE
FROM USER3_HF_FULL_S4663588.ATHLETE1_REPLICA3
WHERE ATHLETEID = 305;

UPDATE USER3_HF_FULL_S4663588.ATHLETE1_REPLICA3
SET ATHLETE1_REPLICA3.CCODE = 'AUS'
WHERE ATHLETE1_REPLICA3.ATHLETEID = 305;

SELECT CCODE
FROM USER3_HF_FULL_S4663588.ATHLETE1_REPLICA3
WHERE ATHLETEID = 305;
```

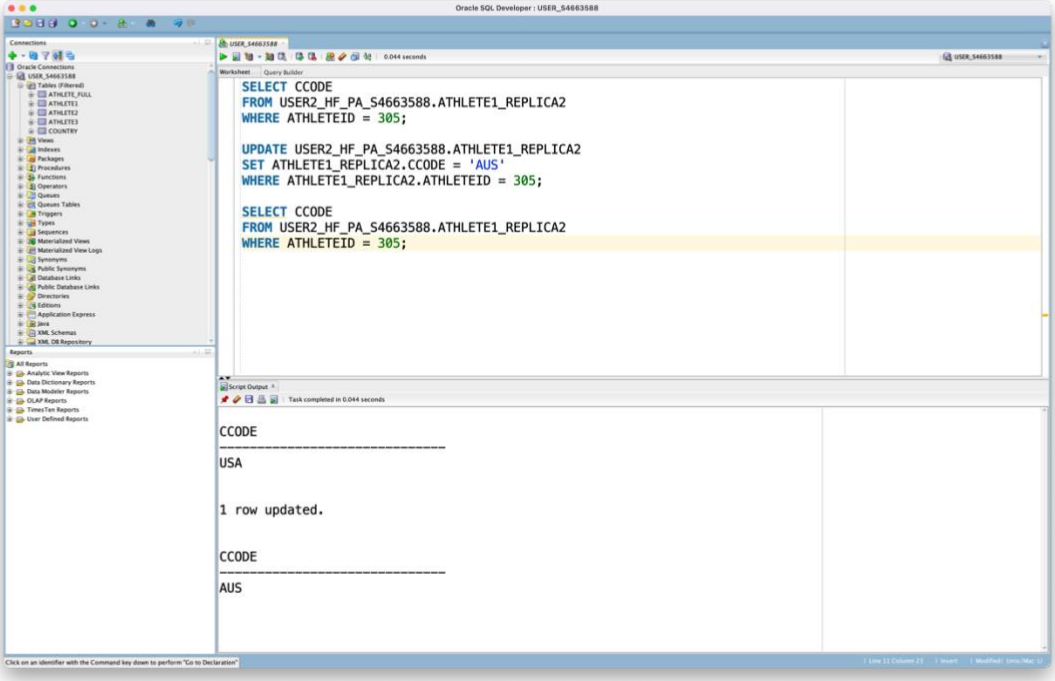
```
CCODE
------------------------------
USA


1 row updated.


CCODE
------------------------------
AUS
```

Due to Partial Replication, more than one site may have a copy of this fragment, but not all of them. Therefore, only part of site (USER1_HF_PA_S4663588 and USER2_HF_PA_S4663588) needs to be modified.



```sql
SELECT CCODE
FROM USER1_HF_PA_S4663588.ATHLETE1_REPLICA1
WHERE ATHLETEID = 305;

UPDATE USER1_HF_PA_S4663588.ATHLETE1_REPLICA1
SET ATHLETE1_REPLICA1.CCODE = 'AUS'
WHERE ATHLETE1_REPLICA1.ATHLETEID = 305;

SELECT CCODE
FROM USER1_HF_PA_S4663588.ATHLETE1_REPLICA1
WHERE ATHLETEID = 305;
```

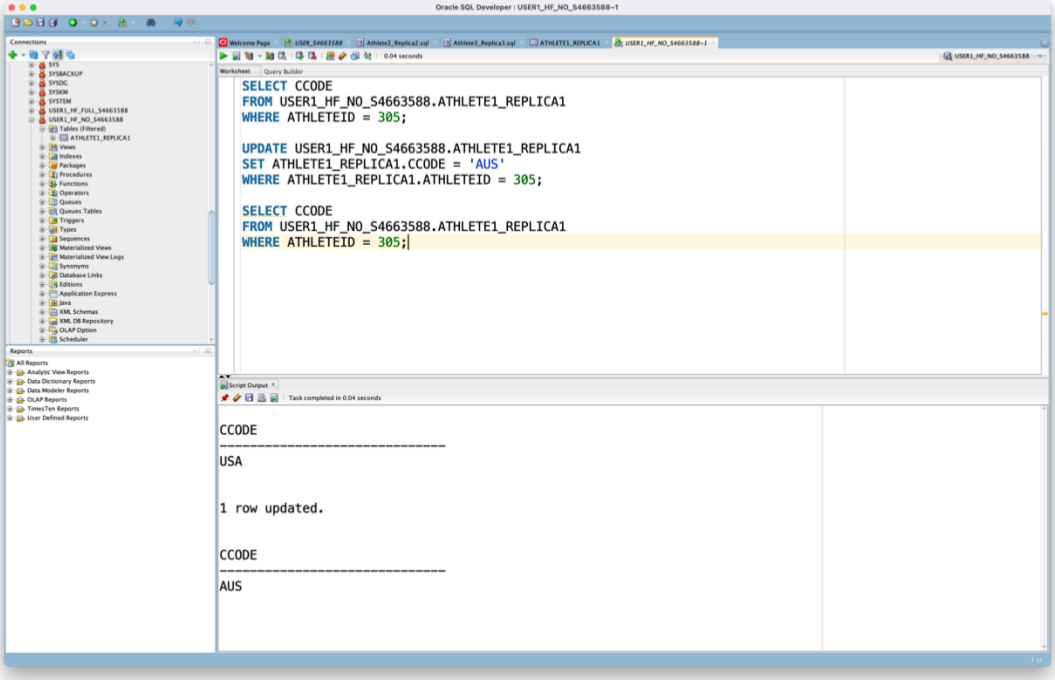
```
CCODE
------------------------------
USA


1 row updated.


CCODE
------------------------------
AUS
```



```
SELECT CCODE
FROM USER2_HF_PA_S4663588.ATHLETE1_REPLICA2
WHERE ATHLETEID = 305;

UPDATE USER2_HF_PA_S4663588.ATHLETE1_REPLICA2
SET ATHLETE1_REPLICA2.CCODE = 'AUS'
WHERE ATHLETE1_REPLICA2.ATHLETEID = 305;

SELECT CCODE
FROM USER2_HF_PA_S4663588.ATHLETE1_REPLICA2
WHERE ATHLETEID = 305;
```

```
CCODE
------------------------------
USA


1 row updated.


CCODE
------------------------------
AUS
```

Due to No Replication, Each fragment will be a relation located on only one site. Therefore, only one site (USER1_HF_NO_S4663588) needs to be modified.



```
SELECT CCODE
FROM USER1_HF_NO_S4663588.ATHLETE1_REPLICA1
WHERE ATHLETEID = 305;

UPDATE USER1_HF_NO_S4663588.ATHLETE1_REPLICA1
SET ATHLETE1_REPLICA1.CCODE = 'AUS'
WHERE ATHLETE1_REPLICA1.ATHLETEID = 305;

SELECT CCODE
FROM USER1_HF_NO_S4663588.ATHLETE1_REPLICA1
WHERE ATHLETEID = 305;
```

```
CCODE
------------------------------
USA


1 row updated.


CCODE
------------------------------
AUS
```

TASK 3

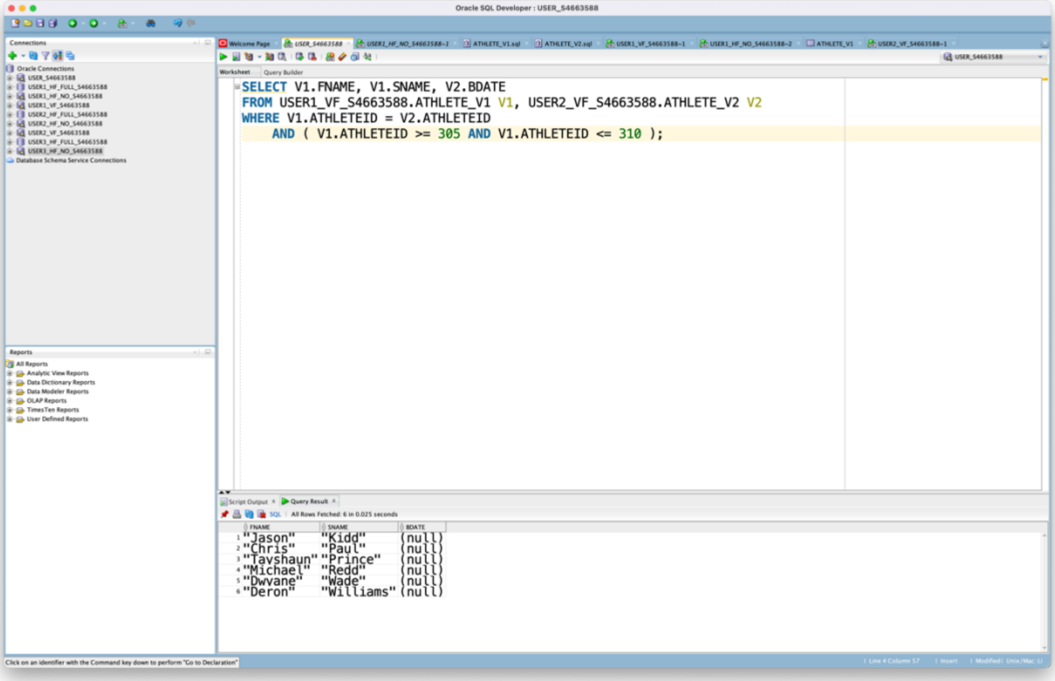
Answer:

Code:

```
SELECT V1.FNAME, V1.SNAME, V2.BDATE
FROM USER1_VF_S4663588.ATHLETE_V1 V1, USER2_VF_S4663588.ATHLETE_V2 V2
WHERE V1.ATHLETEID = V2.ATHLETEID
    AND ( V1.ATHLETEID >= 305 AND V1.ATHLETEID <= 310 );
```

Result:

	FNAME	SNAME	BDATE
1	“Jason”	“Kidd”	(null)
2	“Chris”	“Paul”	(null)
3	“Tayshaun”	“Prince”	(null)
4	“Michael”	“Redd”	(null)
5	“Dwyane”	“Wade”	(null)
6	“Deron”	“Williams”	(null)



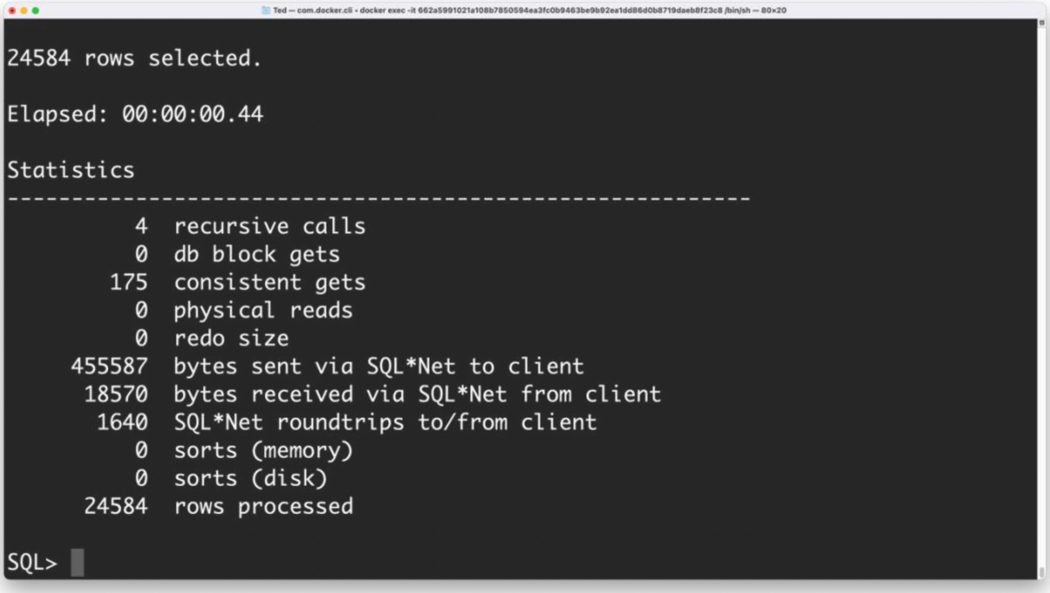
```
SELECT V1.FNAME, V1.SNAME, V2.BDATE
FROM USER1_VF_S4663588.ATHLETE_V1 V1, USER2_VF_S4663588.ATHLETE_V2 V2
WHERE V1.ATHLETEID = V2.ATHLETEID
    AND ( V1.ATHLETEID >= 305 AND V1.ATHLETEID <= 310 );
```

```
  FNAME      SNAME      BDATE
1 "Jason"    "Kidd"     (null)
2 "Chris"    "Paul"     (null)
3 "Tayshaun" "Prince"   (null)
4 "Michael"  "Redd"     (null)
5 "Dwyane"   "Wade"     (null)
6 "Deron"    "Williams" (null)
```

Task 4

Step one

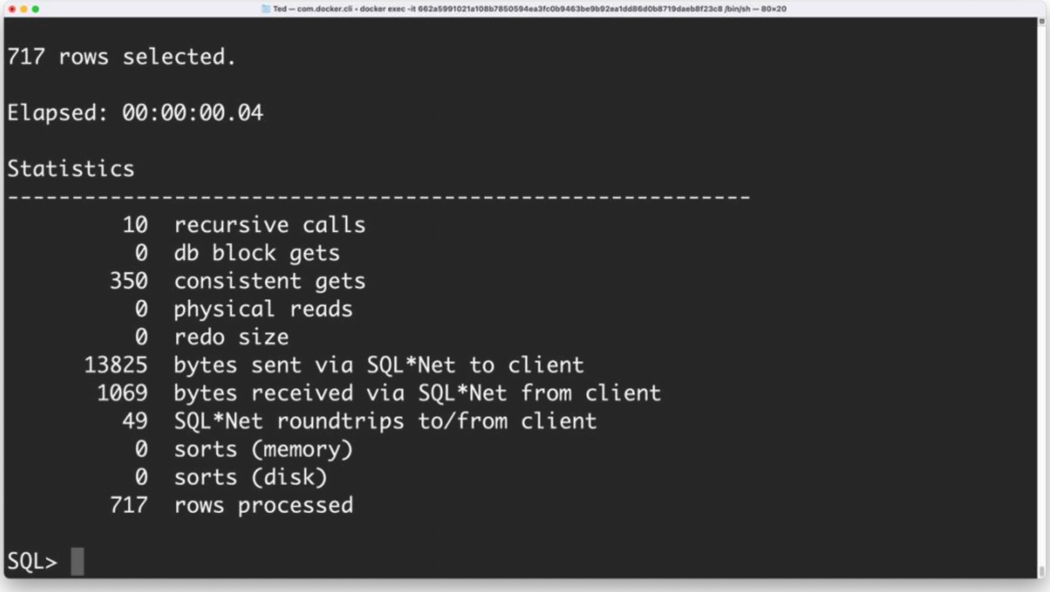
```
SELECT DISTINCT(AthleteID)
FROM "USER1_VF_S4663588"."ATHLETE_V1";
```


Ted — com.docker.cli ◂ docker exec -it 662a5991021a108b7850594ea3fc0b9463be9b92ea1dd86d0b8719daeb8f23c8 /bin/sh — 80×20

```
24584 rows selected.

Elapsed: 00:00:00.44

Statistics
----------------------------------------------------------
          4  recursive calls
          0  db block gets
        175  consistent gets
          0  physical reads
          0  redo size
     455587  bytes sent via SQL*Net to client
      18570  bytes received via SQL*Net from client
       1640  SQL*Net roundtrips to/from client
          0  sorts (memory)
          0  sorts (disk)
      24584  rows processed

SQL>
```

Step two

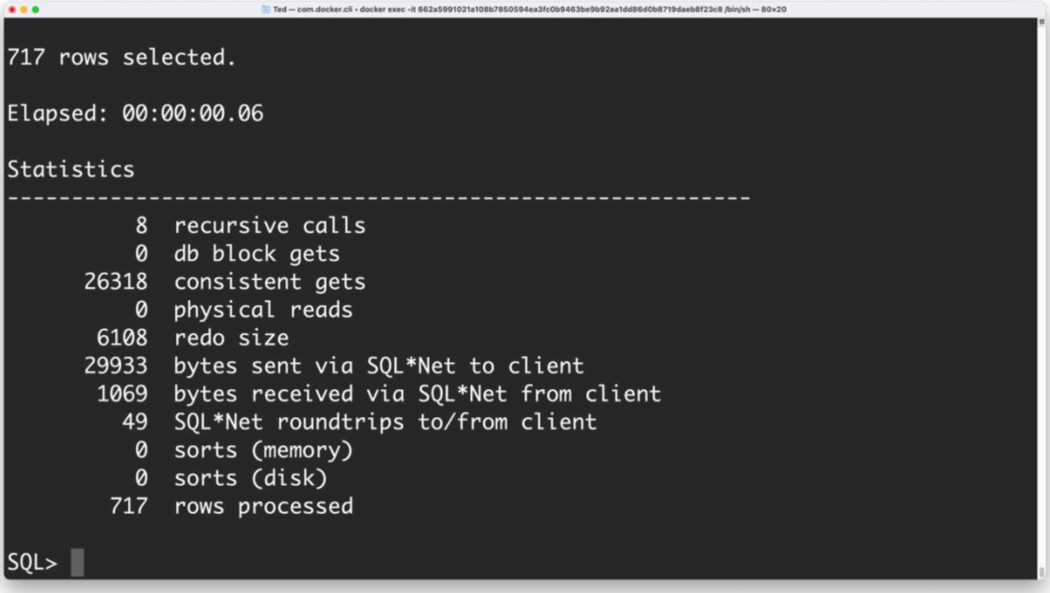
```
SELECT c.BDate, c.CCode, c.SportID
FROM "USER2_VF_S4663588"."ATHLETE_V2" c
WHERE c.CCODE = 'AUS' AND c.AthleteID IN
    (SELECT DISTINCT(AthleteID)
    FROM "USER1_VF_S4663588"."ATHLETE_V1");
```


Ted — com.docker.cli ◂ docker exec -it 662a5991021a108b7850594ea3fc0b9463be9b92ea1dd86d0b8719daeb8f23c8 /bin/sh — 80×20

```
717 rows selected.

Elapsed: 00:00:00.04

Statistics
----------------------------------------------------------
         10  recursive calls
          0  db block gets
        350  consistent gets
          0  physical reads
          0  redo size
      13825  bytes sent via SQL*Net to client
       1069  bytes received via SQL*Net from client
         49  SQL*Net roundtrips to/from client
          0  sorts (memory)
          0  sorts (disk)
        717  rows processed

SQL>
```

Step three

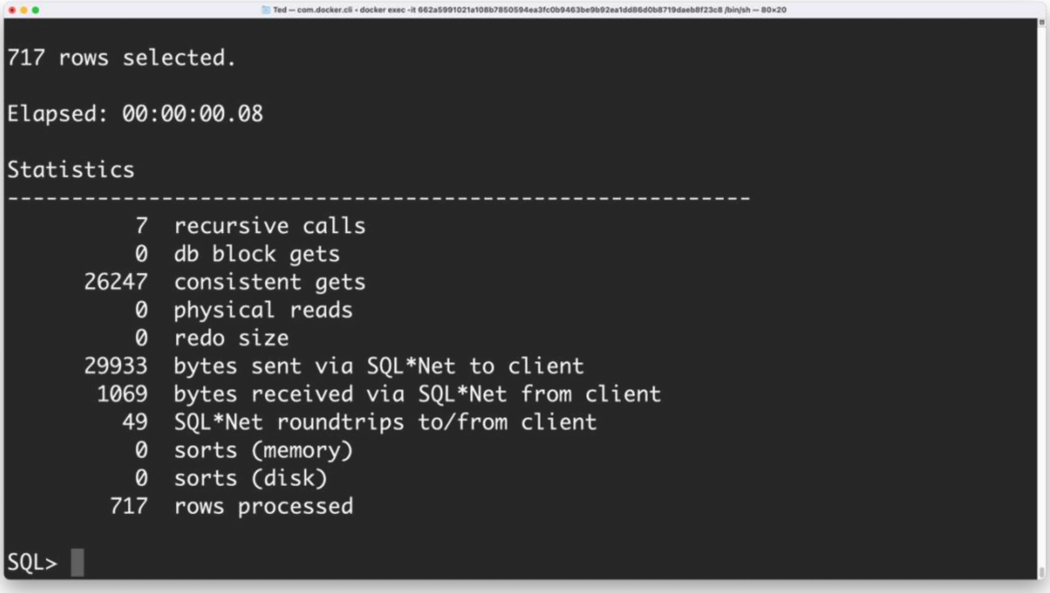
```
SELECT b.AthleteID, b.FName, b.SName, c.BDate, c.CCode, c.SportID
FROM "USER1_VF_S4663588"."ATHLETE_V1" b, (
    SELECT c.AthleteID, c.BDate, c.CCode, c.SportID
    FROM "USER2_VF_S4663588"."ATHLETE_V2" c
    WHERE c.CCODE = 'AUS' AND c.AthleteID IN (
        SELECT DISTINCT(AthleteID)
        FROM "USER1_VF_S4663588"."ATHLETE_V1"
    )
) c
WHERE b.AthleteID = c.AthleteID;
```


Ted — com.docker.cli • docker exec -it 662a5991021a108b7850594ea3fc0b9463be9b92ea1dd86d0b8719daeb8f23c8 /bin/sh — 80×20

```
717 rows selected.

Elapsed: 00:00:00.06

Statistics
----------------------------------------------------------
          8  recursive calls
          0  db block gets
      26318  consistent gets
          0  physical reads
       6108  redo size
      29933  bytes sent via SQL*Net to client
       1069  bytes received via SQL*Net from client
         49  SQL*Net roundtrips to/from client
          0  sorts (memory)
          0  sorts (disk)
        717  rows processed

SQL> 
```

Inner-join:

```
SELECT
    b.AthleteID, b.FName, b.SName, c.BDate, c.CCode, c.SportID
FROM
    "USER1_VF_S4663588"."ATHLETE_V1" b, "USER2_VF_S4663588"."ATHLETE_V2" c
WHERE
    b.AthleteID= c.AthleteID AND c.CCODE='AUS';
```



Ted — com.docker.cli ◂ docker exec -it 662a5991021a108b7850594ea3fc0b9463be9b92ea1dd86d0b8719daeb8f23c8 /bin/sh — 80×20

```
717 rows selected.

Elapsed: 00:00:00.08

Statistics
----------------------------------------------------------
          7  recursive calls
          0  db block gets
      26247  consistent gets
          0  physical reads
          0  redo size
      29933  bytes sent via SQL*Net to client
       1069  bytes received via SQL*Net from client
         49  SQL*Net roundtrips to/from client
          0  sorts (memory)
          0  sorts (disk)
        717  rows processed

SQL> 
```

Therefore, the transmission cost of the semi-join plan is 455587(step 1) + 13825(step 2) = 469412. By decomposing the query into a step-by-step plan, we calculate that the final cost is 469412, but the inner-join cost is 29933. Therefore, we conclude that the cost of the semi-join is higher than that of the inner-join in this example.

Code Appendix

Task 1:

(1)

```
SELECT COUNT(*)
FROM ATHLETE2
WHERE CCODE = 'AUS';
```

(2)

```
SELECT SPORTID AS Sport_ID, COUNT(*) AS Count
FROM ATHLETE3
WHERE CCODE = 'RUS'
GROUP BY SPORTID;
```

(3)

```
CREATE TABLE ATHLETE_FULL(
	ATHLETEID NUMBER NOT NULL,
	FNAME VARCHAR2(30 BYTE),
	SNAME VARCHAR2(30 BYTE),
	BDATE VARCHAR2(30 BYTE),
	CCODE VARCHAR2(30 BYTE),
	SPORTID NUMBER
);

INSERT INTO ATHLETE_FULL
SELECT * FROM ATHLETE1
UNION ALL
SELECT * FROM ATHLETE2
UNION ALL
SELECT * FROM ATHLETE3;

SELECT COUNT(ATHLETE_FULL.ATHLETEID)
FROM ATHLETE_FULL, COUNTRY
WHERE ATHLETE_FULL.CCODE = COUNTRY.CCODE
	AND COUNTRY.CONTINENT = 'EU';
```

Task 2:

```
/*Change Full Replication code*/

UPDATE USER1_HF_FULL_S4663588.ATHLETE1_REPLICA1
SET ATHLETE1_REPLICA1.CCODE = 'AUS'
WHERE ATHLETE1_REPLICA1.ATHLETEID = 305;

UPDATE USER2_HF_FULL_S4663588.ATHLETE1_REPLICA2
SET ATHLETE1_REPLICA2.CCODE = 'AUS'
WHERE ATHLETE1_REPLICA2.ATHLETEID = 305;

UPDATE USER3_HF_FULL_S4663588.ATHLETE1_REPLICA3
SET ATHLETE1_REPLICA3.CCODE = 'AUS'
WHERE ATHLETE1_REPLICA3.ATHLETEID = 305;


/*Change Partial Replication code*/

UPDATE USER1_HF_PA_S4663588.ATHLETE1_REPLICA1
SET ATHLETE1_REPLICA1.CCODE = 'AUS'
WHERE ATHLETE1_REPLICA1.ATHLETEID = 305;

UPDATE USER2_HF_PA_S4663588.ATHLETE1_REPLICA2
SET ATHLETE1_REPLICA2.CCODE = 'AUS'
WHERE ATHLETE1_REPLICA2.ATHLETEID = 305;


/*Change No Replication code*/

UPDATE USER1_HF_NO_S4663588.ATHLETE1_REPLICA1
SET ATHLETE1_REPLICA1.CCODE = 'AUS'
WHERE ATHLETE1_REPLICA1.ATHLETEID = 305;
```

Task 3:

```
SELECT V1.FNAME, V1.SNAME, V2.BDATE
FROM USER1_VF_S4663588.ATHLETE_V1 V1, USER2_VF_S4663588.ATHLETE_V2 V2
WHERE V1.ATHLETEID = V2.ATHLETEID
        AND ( V1.ATHLETEID >= 305 AND V1.ATHLETEID <= 310 );
```

Task 4:

```
/*Step one*/

Select distinct(AthleteID) from "USER1_VF_S4663588"."ATHLETE_V1";
```

```
/*Step two*/

Select c.BDate, c.CCode, c.SportID from "USER2_VF_S4663588"."ATHLETE_V2" c
where c.CCODE = 'AUS' and c.AthleteID in (Select distinct(AthleteID) from
"USER1_VF_S4663588"."ATHLETE_V1");
```

```
/*Step three*/

Select b.AthleteID, b.FName, b.SName, c.BDate, c.CCode, c.SportID from
"USER1_VF_S4663588"."ATHLETE_V1" b, (Select c.AthleteID, c.BDate, c.CCode,
c.SportID from "USER2_VF_S4663588"."ATHLETE_V2" c where c.CCODE = 'AUS' and
c.AthleteID in (Select distinct(AthleteID) from "USER1_VF_S4663588"."ATHLETE_V1"))
c where b.AthleteID = c.AthleteID;
```

```
/*inner-join*/
select b.AthleteID, b.FName, b.SName, c.BDate, c.CCode, c.SportID from
"USER1_VF_S4663588"."ATHLETE_V1" b, "USER2_VF_S4663588"."ATHLETE_V2" c
where b.AthleteID= c.AthleteID and c.CCODE='AUS';
```