Homework #5 – Question #1 –Athletes that shouldn't have received a red card

Compute the number of red cards for either loss of contact or bent knee violation that were given by a judge, for a given race =1, for athletes that didn't deserve to be disqualified based on the video evidence.

An athlete deserves to be disqualified based on the video evidence if they had

1) Two observations with greater than or equal to 60 milliseconds loss of contact, or
2) Two observations with a bent knee greater than 0 degree and less than or equal to 175 degree

Include judges first and last name as well as the number of excessive red cards issued

Order the results by last name and first name

Solution :-

```
SELECT
  J.FirstName,
  J.LastName,
  COUNT(*) AS ExcessiveRedCards
FROM
  JudgeCall JC
  JOIN Judge J ON JC.IDJudge = J.IDJudge
WHERE
  JC.IDRace = 1
  AND JC.Color = 'Red'
  AND (
    JC.Infraction = '~'
    OR JC.Infraction = '<'
  )
  AND JC.BibNumber NOT IN (
    SELECT
      V.BibNumber
    FROM
      VideoObservation V
    WHERE
      V.IDRace = 1
    GROUP BY
      V.BibNumber
    HAVING
      SUM(
        CASE
          WHEN V.LOCAverage >= 60 THEN 1
          ELSE 0
        END
      ) = 2
      OR SUM(
        CASE
          WHEN V.KneeAngle > 0
          AND V.KneeAngle <= 175 THEN 1
          ELSE 0
        END
      ) = 2
  )
GROUP BY
  J.FirstName,
  J.LastName
ORDER BY
  J.LastName,
  J.FirstName;
```

Log | 1: JudgeCall [8]

*	FirstName	LastName	ExcessiveRedCards
1	Kit	Fisto	16
2	Qui-Gon	Jinn	10
3	Obi-Wan	Kenobi	7
4	Leia	Organa	12
5	Jeff	Salvage	12
6	Ahsoka	Tano	9
7	Asajj	Ventress	11
8	Mace	Windu	13

Homework #5 – Question #2 – Judges Matching the Video Observations

Write a query that returns the judge name as well as the number of correct red cards they issued based on the video observations for a given race = 1
A red card for loss of contact is correct if there are two observations with greater than or equal to 60 milliseconds loss of contact
OR
A red card for a bent knee is correct if there are two observations with bent knee greater than 0 degree and less than or equal to 175 degrees.
Order the results by last name and first name

Solution:-

```
SELECT j.FirstName, j.LastName, COUNT(*) AS CorrectRedCards
FROM Judge j
INNER JOIN RaceJudge rj ON j.IDJudge = rj.IDJudge
INNER JOIN JudgeCall jc ON rj.IDRace = jc.IDRace AND rj.IDJudge = jc.IDJudge
WHERE jc.IDRace = 1 AND jc.Color = 'Red'
    AND (
        -- Check for correct loss of contact red cards
        (jc.Infraction = '~' AND EXISTS (
            SELECT 1
            FROM VideoObservation vo1
            INNER JOIN VideoObservation vo2 ON vo1.IDRace = vo2.IDRace AND vo1.BibNumber
= vo2.BibNumber
            WHERE vo1.IDRace = 1 AND vo1.BibNumber = jc.BibNumber AND vo1.LOCAverage >=
60
                AND vo2.IDRace = 1 AND vo2.BibNumber = jc.BibNumber AND vo2.LOCAverage >=
60
        ))
        OR
        -- Check for correct bent knee red cards
        (jc.Infraction = '<' AND EXISTS (
            SELECT 1
            FROM VideoObservation vo1
            INNER JOIN VideoObservation vo2 ON vo1.IDRace = vo2.IDRace AND vo1.BibNumber
= vo2.BibNumber
            WHERE vo1.IDRace = 1 AND vo1.BibNumber = jc.BibNumber AND vo1.KneeAngle > 0
AND vo1.KneeAngle <= 175
                AND vo2.IDRace = 1 AND vo2.BibNumber = jc.BibNumber AND vo2.KneeAngle > 0
AND vo2.KneeAngle <= 175
        ))
```

```
    )
GROUP BY j.FirstName, j.LastName
ORDER BY j.LastName, j.FirstName;
```

Explanation :-

Filtering based on Infraction:

The query still filters for red cards (jc.Color = 'RED') and considers only loss of contact and bent knee infractions (jc.Infraction IN ('Loss of Contact ~', 'Bent Knee <')).

Subqueries for Correctness:

Instead of excluding disqualified athletes, the query now uses subqueries to check if there are two observations supporting the red card for each BibNumber:

The first subquery checks for loss of contact red cards. It searches for two observations with LOCAverage greater than or equal to 60 milliseconds for the same athlete (jc.BibNumber) in the same race (vo1.IDRace = 1 and vo2.IDRace = 1).

The second subquery checks for bent knee red cards. It searches for two observations with the bent knee angle greater than 0 and less than or equal to 175 degrees for the same athlete (jc.BibNumber) in the same race (vo1.IDRace = 1 and vo2.IDRace = 1).

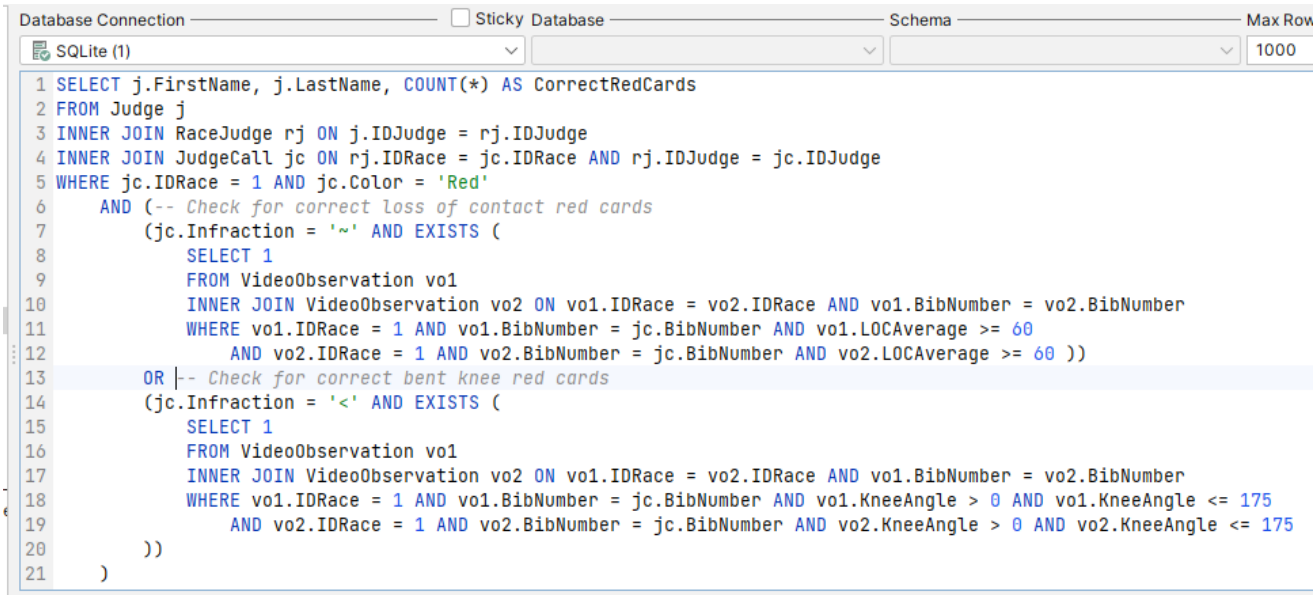
Filtering with EXISTS:

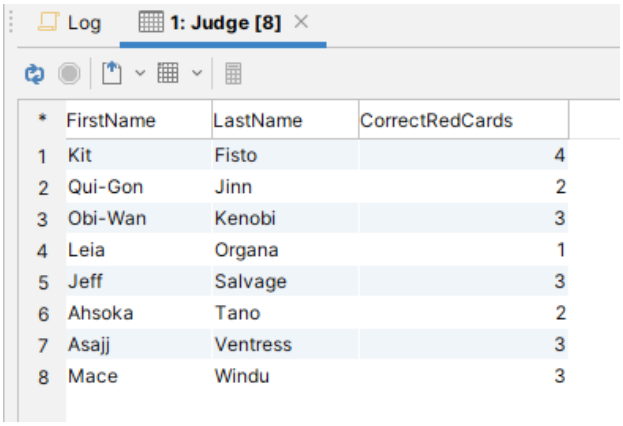
The EXISTS operator checks if the subqueries return at least one row, indicating the red card was supported by the video evidence.

The WHERE clause uses an OR operator to combine the conditions for both types of infractions.

Aggregation and Ordering:

Similar to the previous query, the results are grouped and ordered by the judge's name, with the final count representing the number of correct red cards issued by each judge.





*	FirstName	LastName	CorrectRedCards
1	Kit	Fisto	4
2	Qui-Gon	Jinn	2
3	Obi-Wan	Kenobi	3
4	Leia	Organa	1
5	Jeff	Salvage	3
6	Ahsoka	Tano	2
7	Asajj	Ventress	3
8	Mace	Windu	3

Homework #5 – Question #3 – Judges Missing Red Cards from the Video Observations

Write a query that returns the judge name as well as the number of missed red card they issued based on the video observations for a given race = 1.

A red card for loss of contact is correct if there are two observations with greater than or equal to 60 milliseconds loss of contact

A red card for bent knee is correct if there are two observations with bent knee greater than 0 degrees and less than or equal to 175 degrees.

Order the results by last name and first name

Solution:-

```
-- Homework #5 - Question #3 - Judges Missing Red Cards from the Video Observations
-------------------------------------------------------------------------------
SELECT j.FirstName, j.LastName,
       COUNT(CASE WHEN jc.Infraction = '~' THEN 1 END) AS MissedRedCardLOC,
       COUNT(CASE WHEN jc.Infraction = '<' THEN 1 END) AS MissedRedCardBentKnee
FROM Judge j
INNER JOIN RaceJudge rj ON j.IDJudge = rj.IDJudge
INNER JOIN Bib b ON rj.IDRace = b.IDRace
LEFT JOIN JudgeCall jc ON b.IDRace = jc.IDRace AND b.BibNumber = jc.BibNumber
WHERE b.IDRace = 1
    AND (
        -- Check for missed loss of contact red cards
        (EXISTS (
            SELECT 1
            FROM VideoObservation vo1
            INNER JOIN VideoObservation vo2 ON vo1.IDRace = vo2.IDRace AND vo1.BibNumber
= vo2.BibNumber
            WHERE vo1.IDRace = 1 AND b.BibNumber = vo1.BibNumber AND vo1.LOCAverage >= 60
                AND vo2.IDRace = 1 AND b.BibNumber = vo2.BibNumber AND vo2.LOCAverage >=
60
        ))
        AND jc.IDRace IS NULL OR jc.Color <> 'Red' OR jc.Infraction <> '~'
    )
    OR
    (
        -- Check for missed bent knee red cards
        (EXISTS (
            SELECT 1
            FROM VideoObservation vo1
            INNER JOIN VideoObservation vo2 ON vo1.IDRace = vo2.IDRace AND vo1.BibNumber
= vo2.BibNumber
            WHERE vo1.IDRace = 1 AND b.BibNumber = vo1.BibNumber AND vo1.KneeAngle > 0
AND vo1.KneeAngle <= 175
                AND vo2.IDRace = 1 AND b.BibNumber = vo2.BibNumber AND vo2.KneeAngle > 0
AND vo2.KneeAngle <= 175
        ))
        AND jc.IDRace IS NULL OR jc.Color <> 'Red' OR jc.Infraction <> '<'
    )
GROUP BY j.FirstName, j.LastName
ORDER BY j.LastName, j.FirstName;
```

Database Connection — Sticky Database — Schema

SQLite (1)

```sql
-- Homework #5 - Question #3 - Judges Missing Red Cards from the Video Observations
-------------------------------------------------------------------------------
SELECT j.FirstName, j.LastName,
       COUNT(CASE WHEN jc.Infraction = '~' THEN 1 END) AS MissedRedCardLOC,
       COUNT(CASE WHEN jc.Infraction = '<' THEN 1 END) AS MissedRedCardBentKnee
FROM Judge j
INNER JOIN RaceJudge rj ON j.IDJudge = rj.IDJudge
INNER JOIN Bib b ON rj.IDRace = b.IDRace
LEFT JOIN JudgeCall jc ON b.IDRace = jc.IDRace AND b.BibNumber = jc.BibNumber
WHERE b.IDRace = 1
    AND ( -- Check for missed loss of contact red cards
        (EXISTS (
            SELECT 1
            FROM VideoObservation vo1
            INNER JOIN VideoObservation vo2 ON vo1.IDRace = vo2.IDRace AND vo1.BibNumber = vo2.BibNumber
            WHERE vo1.IDRace = 1 AND b.BibNumber = vo1.BibNumber AND vo1.LOCAverage >= 60
                AND vo2.IDRace = 1 AND b.BibNumber = vo2.BibNumber AND vo2.LOCAverage >= 60
        ))
        AND jc.IDRace IS NULL OR jc.Color <> 'Red' OR jc.Infraction <> '~'
    )
    OR
```

Log | 1: Judge [8]

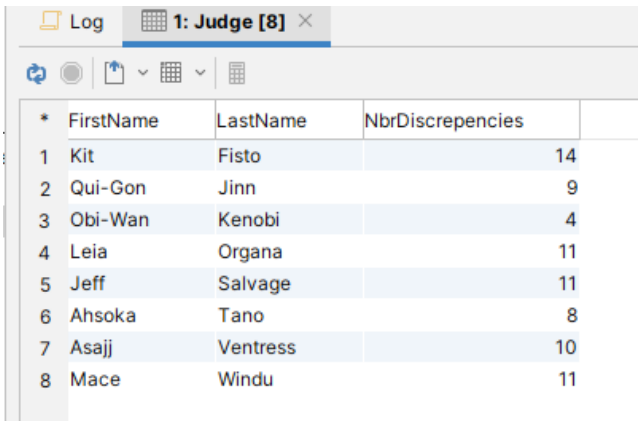
*	FirstName	LastName	MissedRedCardLOC	MissedRedCardBentKnee
1	Kit	Fisto	399	216
2	Qui-Gon	Jinn	399	216
3	Obi-Wan	Kenobi	285	191
4	Leia	Organa	399	216
5	Jeff	Salvage	278	190
6	Ahsoka	Tano	278	190
7	Asajj	Ventress	399	216
8	Mace	Windu	285	191

Homework #5 – Question #4 – Judges Discrepancy from the Video Observations

Write a query to return the number of red cards a judge gave for the wrong reason (either a ~ when it should be < or visa versa). List the judge name, and number of discrepancies. Sort the results by judge name (last name, first name)

Solution:-

```
-- Homework #5 - Question #4 - Judges Discrepancy from the Video Observations
-- Write a query to return the number of red cards a judge gave for the wrong reason
-- (either a ~ when it should be < or visa versa). List the judge name, and number of
discrepancies.
-- Sort the results by judge name (last name, first name)
-------------------------------------------------------------------------------
SELECT j.FirstName, j.LastName, COUNT(*) AS NbrDiscrepencies
FROM Judge j
INNER JOIN JudgeCall jc ON j.IDJudge = jc.IDJudge
WHERE jc.IDRace = 1 AND jc.Color = 'Red'
AND (
  -- Check for red cards for bent knee where video suggests LOC
  (jc.Infraction = '<' AND NOT EXISTS (
    SELECT 1
    FROM VideoObservation vo
    WHERE vo.IDRace = 1 AND vo.BibNumber = jc.BibNumber AND vo.LOCAverage >= 60
  ))
  OR
  -- Check for red cards for LOC where video suggests bent knee
  (jc.Infraction = '~' AND NOT EXISTS (
    SELECT 1
    FROM VideoObservation vo1
    INNER JOIN VideoObservation vo2 ON vo1.IDRace = vo2.IDRace AND vo1.BibNumber =
vo2.BibNumber
    WHERE vo1.IDRace = 1 AND vo1.BibNumber = jc.BibNumber AND vo1.KneeAngle > 0 AND
vo1.KneeAngle <= 175
                AND vo2.IDRace = 1 AND vo2.BibNumber = jc.BibNumber AND vo2.KneeAngle > 0
AND vo2.KneeAngle <= 175
  ))
)
GROUP BY j.FirstName, j.LastName
ORDER BY j.LastName, j.FirstName;
```



Log | 1: Judge [8]

*	FirstName	LastName	NbrDiscrepencies
1	Kit	Fisto	14
2	Qui-Gon	Jinn	9
3	Obi-Wan	Kenobi	4
4	Leia	Organa	11
5	Jeff	Salvage	11
6	Ahsoka	Tano	8
7	Asajj	Ventress	10
8	Mace	Windu	11

Homework #5 – Question #5 – Correct Non Calls

Judge must also not make calls when the walker are legal
A walker is legal if they do not have 2 instances of loss of contact on the video observations
A walker is legal if they do not have 2 instances of bent knee on the video observations
A walker is legal if they have 1 instance of loss of contact and 1 instance of bent knee on the video observations
The query should return the judge names and the number on correct non calls for race =1
Sort the results by judge name (lastname, firstname)

```
-- HW5  #5
-- Judge must also not make calls when the walker are legal
-- A walker is legal if they do not have 2 instances of loss of contact on the video
observations
-- A walker is legal if they do not have 2 instances of bent knee on the video
observations
-- A walker is legal if they have 1 instance of loss of contact and 1 instance of bent
knee on the video observations
-- The query should return the judge names and the number on correct non calls for race
=1
-- Sort the results by judge name (lastname, firstname)

SELECT j.FirstName, j.LastName, COUNT(*) AS NbrCorrectNonCalls
FROM JudgeCall jc
JOIN Judge j ON jc.IDJudge = j.IDJudge
WHERE jc.IDRace = 1 AND jc.Color = 'Yellow' AND
(
  -- Check for yellow cards not requiring a penalty based on legal walker criteria
  BibNumber IN (
    SELECT vo.BibNumber
    FROM VideoObservation vo
    WHERE vo.IDRace = 1 AND
      (
        -- Use subqueries for counting within conditional statements
        (SELECT COUNT(*) FROM (SELECT 1 FROM videoObservation WHERE KneeAngle >0 AND
KneeAngle <= 175) AS cnt) < 2 AND
        (SELECT COUNT(*) FROM (SELECT 1 FROM videoObservation WHERE LOCAverage < 60) AS
cnt) < 2 OR
        (SELECT COUNT(*) FROM (SELECT 1 FROM videoObservation WHERE KneeAngle >0 AND
KneeAngle <=175) AS cnt) = 1 AND
        (SELECT COUNT(*) FROM (SELECT 1 FROM videoObservation WHERE LOCAverage < 60) AS
cnt) = 1
      )
  )
)
GROUP BY j.IDJudge, j.FirstName, j.LastName
ORDER BY j.LastName, j.FirstName;
```

Homework #5 – Question #6 – Compute a Total Judge Score

Create a single query that computes a score for each judge at a given race.
The judge score is a weighted sum of each value returned from the previous queries.
The result set should include the name of the judge, the values returned from the previous queries and the total score
The total score follows the following formula.
Correct RedCards *2 + CorrectNonCalls * .5 – ExcessiveRedCards – MissedLOCRed *2 - MissedBentRed *2 - NbrDiscrepancies

Sort the result by judge name (last name, first name)

```
SELECT j.FirstName,
j.LastName,NumCorrectRedCards,NbrCorrectNonCalls,NumExcessiveRedCards,
       NumMissedLOCCards,NumMissedBentKneeCards,NbrDiscrepancies,
       CRC.NumCorrectRedCards * 2 + CNC.NbrCorrectNonCalls * 0.5 -
ERC.NumExcessiveRedCards -
       MLOC.NumMissedLOCCards * 2 - MBK.NumMissedBentKneeCards * 2 - ND.NbrDiscrepancies
AS score
FROM Judge j

LEFT JOIN (
  -------------- Query for correct red cards --------------------------
SELECT jc.IDJudge, COUNT(*) AS NumCorrectRedCards
FROM Judge j
INNER JOIN RaceJudge rj ON j.IDJudge = rj.IDJudge
INNER JOIN JudgeCall jc ON rj.IDRace = jc.IDRace AND rj.IDJudge = jc.IDJudge
WHERE jc.IDRace = 1 AND jc.Color = 'Red'
    AND (
        -- Check for correct loss of contact red cards
        (jc.Infraction = '~' AND EXISTS (
            SELECT 1
            FROM VideoObservation vo1
            INNER JOIN VideoObservation vo2 ON vo1.IDRace = vo2.IDRace AND vo1.BibNumber
= vo2.BibNumber
            WHERE vo1.IDRace = 1 AND vo1.BibNumber = jc.BibNumber AND vo1.LOCAverage >=
60
                AND vo2.IDRace = 1 AND vo2.BibNumber = jc.BibNumber AND vo2.LOCAverage >=
60
        ))
        OR
        -- Check for correct bent knee red cards
        (jc.Infraction = '<' AND EXISTS (
            SELECT 1
            FROM VideoObservation vo1
            INNER JOIN VideoObservation vo2 ON vo1.IDRace = vo2.IDRace AND vo1.BibNumber
= vo2.BibNumber
            WHERE vo1.IDRace = 1 AND vo1.BibNumber = jc.BibNumber AND vo1.KneeAngle > 0
AND vo1.KneeAngle <= 175
                AND vo2.IDRace = 1 AND vo2.BibNumber = jc.BibNumber AND vo2.KneeAngle > 0
AND vo2.KneeAngle <= 175
        ))
    )
GROUP BY jc.IDJudge
ORDER BY jc.IDJudge

  -------------- Specify the judge's ID from the JudgeCall table
) CRC ON j.IDJudge = CRC.IDJudge
LEFT JOIN (
  -- Query for correct non-calls (same as previous)
  SELECT j.IDJudge, COUNT(*) AS NbrCorrectNonCalls
  FROM JudgeCall jc
  JOIN Judge j ON jc.IDJudge = j.IDJudge
  WHERE jc.IDRace = 1 AND jc.Color = 'Yellow'
  GROUP BY j.IDJudge
) CNC ON j.IDJudge = CNC.IDJudge
LEFT JOIN (
  -- Query for excessive red cards
  SELECT jc.IDJudge,COUNT(*) AS NumExcessiveRedCards
FROM
  JudgeCall JC
  JOIN Judge J ON JC.IDJudge = J.IDJudge
WHERE
  JC.IDRace = 1 AND JC.Color = 'Red' AND (JC.Infraction = '~' OR JC.Infraction = '<' )
```

```
  AND JC.BibNumber NOT IN ( SELECT V.BibNumber FROM VideoObservation V
    WHERE V.IDRace = 1
    GROUP BY V.BibNumber
    HAVING SUM(
       CASE
          WHEN V.LOCAverage >= 60 THEN 1
          ELSE 0
        END
      ) = 2
      OR SUM(
        CASE
          WHEN V.KneeAngle > 0
          AND V.KneeAngle <= 175 THEN 1
          ELSE 0
        END
      ) = 2
  )
GROUP BY jc.IDJudge
ORDER BY jc.IDJudge

  -- Specify the judge's ID from the JudgeCall table
) ERC ON j.IDJudge = ERC.IDJudge
LEFT JOIN (
  -------------------------------- Query for missed LOC red cards--------------------
  --SELECT jc.IDJudge, COUNT(*) AS NumMissedLOCCards
  --FROM JudgeCall jc
  ------------------------------------------------------------------------------------
 SELECT jc.IDJudge, COUNT(*) AS NumMissedLOCCards,
       COUNT(CASE WHEN jc.Infraction = '~' THEN 1 END) AS MissedRedCardLOC,
       COUNT(CASE WHEN jc.Infraction = '<' THEN 1 END) AS MissedRedCardBentKnee
FROM Judge j
INNER JOIN RaceJudge rj ON j.IDJudge = rj.IDJudge
INNER JOIN Bib b ON rj.IDRace = b.IDRace
LEFT JOIN JudgeCall jc ON b.IDRace = jc.IDRace AND b.BibNumber = jc.BibNumber
WHERE b.IDRace = 1
    AND (
        -- Check for missed loss of contact red cards
        (EXISTS (
            SELECT 1
            FROM VideoObservation vo1
            INNER JOIN VideoObservation vo2 ON vo1.IDRace = vo2.IDRace AND vo1.BibNumber
= vo2.BibNumber
            WHERE vo1.IDRace = 1 AND b.BibNumber = vo1.BibNumber AND vo1.LOCAverage >= 60
                AND vo2.IDRace = 1 AND b.BibNumber = vo2.BibNumber AND vo2.LOCAverage >=
60
        ))
        AND jc.IDRace IS NULL OR jc.Color <> 'Red' OR jc.Infraction <> '~'
    )
    OR
    (
        -- Check for missed bent knee red cards
        (EXISTS (
            SELECT 1
            FROM VideoObservation vo1
            INNER JOIN VideoObservation vo2 ON vo1.IDRace = vo2.IDRace AND vo1.BibNumber
= vo2.BibNumber
            WHERE vo1.IDRace = 1 AND b.BibNumber = vo1.BibNumber AND vo1.KneeAngle > 0
AND vo1.KneeAngle <= 175
                AND vo2.IDRace = 1 AND b.BibNumber = vo2.BibNumber AND vo2.KneeAngle > 0
AND vo2.KneeAngle <= 175
        ))
        AND jc.IDRace IS NULL OR jc.Color <> 'Red' OR jc.Infraction <> '<'
```

```
    )
GROUP BY jc.IDJudge
ORDER BY jc.IDJudge

  -------------- Specify the judge's ID from the JudgeCall table---------------
) MLOC ON j.IDJudge = MLOC.IDJudge
LEFT JOIN (
  -- Query for missed bent knee red cards
  SELECT jc.IDJudge, COUNT(*) AS NumMissedBentKneeCards
  FROM JudgeCall jc
  WHERE jc.IDRace = 1 AND jc.Color = 'Yellow' AND NOT EXISTS (
    SELECT 1
    FROM VideoObservation vo1
    INNER JOIN VideoObservation vo2 ON vo1.IDRace = vo2.IDRace AND vo1.BibNumber =
vo2.BibNumber
    WHERE vo1.IDRace = jc.IDRace AND vo1.BibNumber = jc.BibNumber AND vo1.KneeAngle > 0
AND vo1.KneeAngle <= 175
                AND vo2.IDRace = jc.IDRace AND vo2.BibNumber = jc.BibNumber AND
vo2.KneeAngle > 0 AND vo2.KneeAngle <= 175
  )
  GROUP BY jc.IDJudge  -- Specify the judge's ID from the JudgeCall table
) MBK ON j.IDJudge = MBK.IDJudge
LEFT JOIN (
  -- Query for number of discrepancies (any judge call that doesn't match video
observations)
SELECT j.IDJudge, COUNT(*) AS NbrDiscrepancies
FROM Judge j
INNER JOIN JudgeCall jc ON j.IDJudge = jc.IDJudge
WHERE jc.IDRace = 1 AND jc.Color = 'Red'
AND (
  -- Check for red cards for bent knee where video suggests LOC
  (jc.Infraction = '<' AND NOT EXISTS (
    SELECT 1
    FROM VideoObservation vo
    WHERE vo.IDRace = 1 AND vo.BibNumber = jc.BibNumber AND vo.LOCAverage >= 60
  ))
  OR
  -- Check for red cards for LOC where video suggests bent knee
  (jc.Infraction = '~' AND NOT EXISTS (
    SELECT 1
    FROM VideoObservation vo1
    INNER JOIN VideoObservation vo2 ON vo1.IDRace = vo2.IDRace AND vo1.BibNumber =
vo2.BibNumber
    WHERE vo1.IDRace = 1 AND vo1.BibNumber = jc.BibNumber AND vo1.KneeAngle > 0 AND
vo1.KneeAngle <= 175
                AND vo2.IDRace = 1 AND vo2.BibNumber = jc.BibNumber AND vo2.KneeAngle > 0
AND vo2.KneeAngle <= 175
  ))
)
GROUP BY j.IDJudge
ORDER BY j.IDJudge
-------------------------------------------------------------------------------------
)ND ON j.IDJudge = ND.IDJudge
order by j.lastname,j.firstname
```

Log | 1: Judge [8]

*	FirstName	LastName	NumCorrectRedCards	NbrCorrectNonCalls	NumExcessiveRedCards	NumMissedLOCCards
1	Kit	Fisto	4	32	16	
2	Qui-Gon	Jinn	2	29	10	
3	Obi-Wan	Kenobi	3	14	7	
4	Leia	Organa	1	33	12	
5	Jeff	Salvage	3	30	12	
6	Ahsoka	Tano	2	25	9	
7	Asajj	Ventress	3	35	11	
8	Mace	Windu	3	40	13	