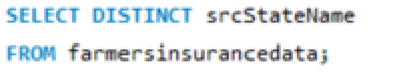
Farmers Insurance Analysis - Assignment

Group Member: Ankit Bougal & Utkarsh Bora

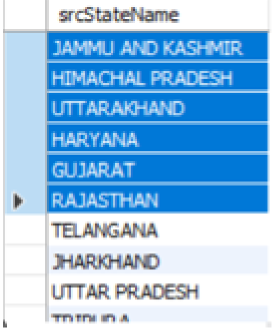
Q1. Retrieve the names of all states (srcStateName) from the dataset.

Input



```sql
SELECT DISTINCT srcStateName
FROM farmersinsurancedata;
```

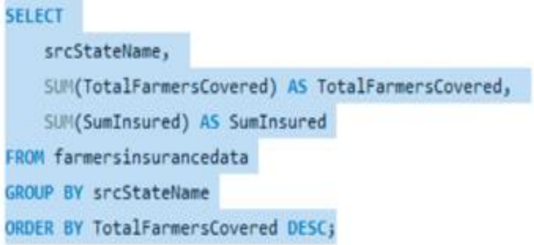
Output



```
srcStateName
JAMMU AND KASHMIR
HIMACHAL PRADESH
UTTARAKHAND
HARYANA
GUJARAT
RAJASTHAN
TELANGANA
JHARKHAND
UTTAR PRADESH
TRIPURA
```

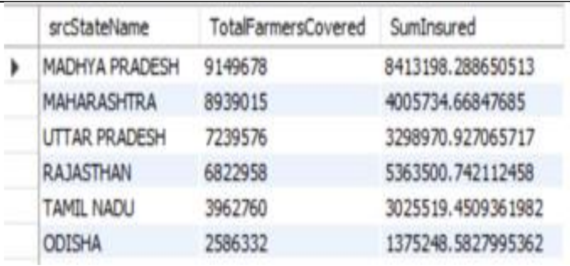
Q2. Retrieve the total number of farmers covered (TotalFarmersCovered) and the sum insured (SumInsured) for each state (srcStateName), ordered by TotalFarmersCovered in descending order.

Input



```sql
SELECT
    srcStateName,
    SUM(TotalFarmersCovered) AS TotalFarmersCovered,
    SUM(SumInsured) AS SumInsured
FROM farmersinsurancedata
GROUP BY srcStateName
ORDER BY TotalFarmersCovered DESC;
```

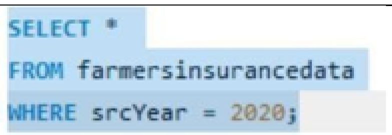
OutPut



```
srcStateName     TotalFarmersCovered   SumInsured
MADHYA PRADESH   9149678               8413198.288650513
MAHARASHTRA      8939015               4005734.66847685
UTTAR PRADESH    7239576               3298970.927065717
RAJASTHAN        6822958               5363500.742112458
TAMIL NADU       3962760               3025519.4509361982
ODISHA           2586332               1375248.5827995362
```

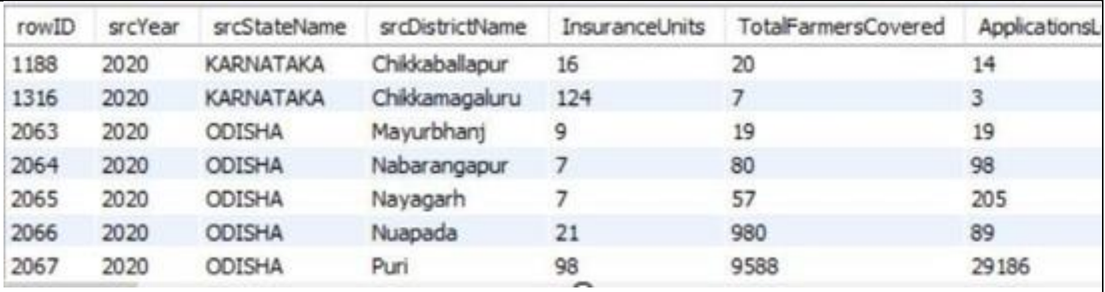
Q3. Retrieve all records where Year is '2020'.

Input



```sql
SELECT *
FROM farmersinsurancedata
WHERE srcYear = 2020;
```

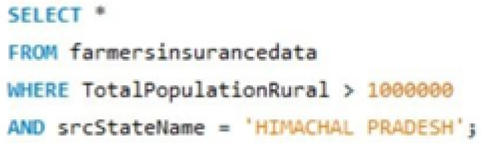
Output



rowID	srcYear	srcStateName	srcDistrictName	InsuranceUnits	TotalFarmersCovered	ApplicationsL
1188	2020	KARNATAKA	Chikkaballapur	16	20	14
1316	2020	KARNATAKA	Chikkamagaluru	124	7	3
2063	2020	ODISHA	Mayurbhanj	9	19	19
2064	2020	ODISHA	Nabarangapur	7	80	98
2065	2020	ODISHA	Nayagarh	7	57	205
2066	2020	ODISHA	Nuapada	21	980	89
2067	2020	ODISHA	Puri	98	9588	29186

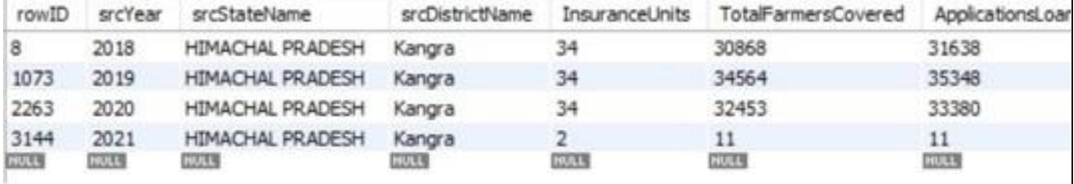
Q4. Retrieve all rows where the TotalPopulationRural is greater than 1 million and the srcStateName is 'HIMACHAL PRADESH'.

Input



```
SELECT *
FROM farmersinsurancedata
WHERE TotalPopulationRural > 1000000
AND srcStateName = 'HIMACHAL PRADESH';
```

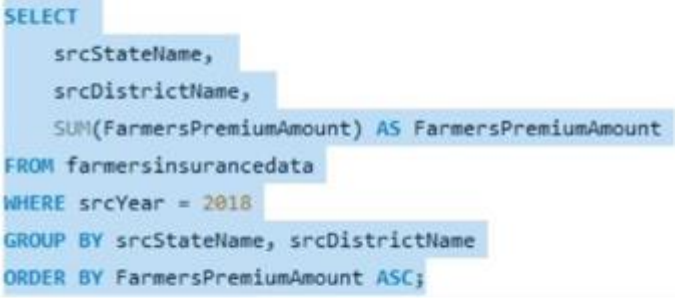
output



rowID	srcYear	srcStateName	srcDistrictName	InsuranceUnits	TotalFarmersCovered	ApplicationsLoar
8	2018	HIMACHAL PRADESH	Kangra	34	30868	31638
1073	2019	HIMACHAL PRADESH	Kangra	34	34564	35348
2263	2020	HIMACHAL PRADESH	Kangra	34	32453	33380
3144	2021	HIMACHAL PRADESH	Kangra	2	11	11
NULL	NULL	NULL	NULL	NULL	NULL	NULL

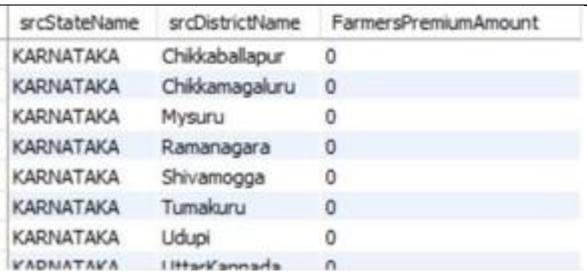
Q5. Retrieve the srcStateName, srcDistrictName, and the sum of FarmersPremiumAmount for each district in the year 2018, and display the results ordered by FarmersPremiumAmount in ascending order.

Input



```
SELECT
    srcStateName,
    srcDistrictName,
    SUM(FarmersPremiumAmount) AS FarmersPremiumAmount
FROM farmersinsurancedata
WHERE srcYear = 2018
GROUP BY srcStateName, srcDistrictName
ORDER BY FarmersPremiumAmount ASC;
```

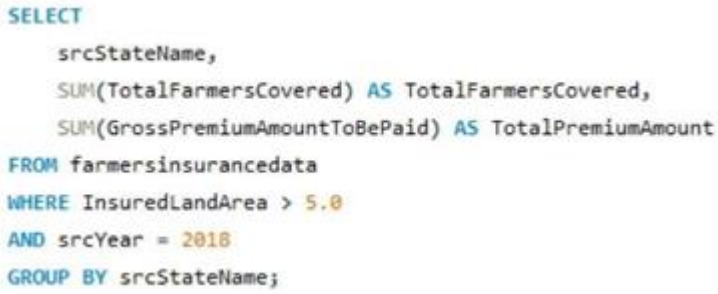
Output



```
srcStateName  srcDistrictName  FarmersPremiumAmount
KARNATAKA     Chikkaballapur   0
KARNATAKA     Chikkamagaluru   0
KARNATAKA     Mysuru           0
KARNATAKA     Ramanagara       0
KARNATAKA     Shivamogga       0
KARNATAKA     Tumakuru         0
KARNATAKA     Udupi            0
KARNATAKA     UttarKannada     0
```

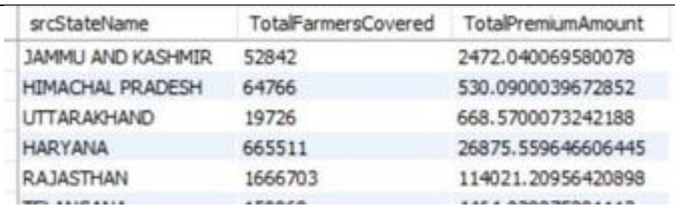
Q6. Retrieve the total number of farmers covered (TotalFarmersCovered) and the sum of premiums (GrossPremiumAmountToBePaid) for each state (srcStateName) where the insured land area (InsuredLandArea) is greater than 5.0 and the Year is 2018.

Input



```
SELECT
    srcStateName,
    SUM(TotalFarmersCovered) AS TotalFarmersCovered,
    SUM(GrossPremiumAmountToBePaid) AS TotalPremiumAmount
FROM farmersinsurancedata
WHERE InsuredLandArea > 5.0
AND srcYear = 2018
GROUP BY srcStateName;
```

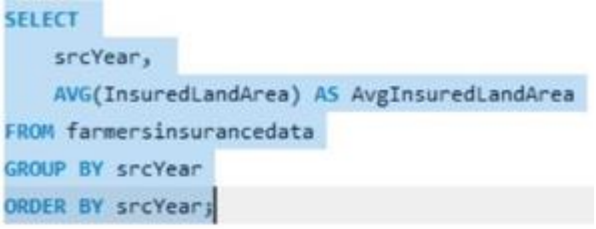
Output



```
srcStateName        TotalFarmersCovered  TotalPremiumAmount
JAMMU AND KASHMIR   52842                2472.040069580078
HIMACHAL PRADESH    64766                530.0900039672852
UTTARAKHAND         19726                668.5700073242188
HARYANA             665511               26875.559646606445
RAJASTHAN           1666703              114021.20956420898
[illegible]
```

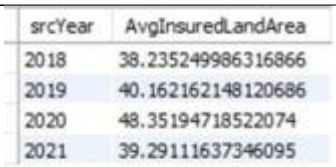
Q7. Calculate the average insured land area (InsuredLandArea) for each year (srcYear).

Input



```
SELECT
    srcYear,
    AVG(InsuredLandArea) AS AvgInsuredLandArea
FROM farmersinsurancedata
GROUP BY srcYear
ORDER BY srcYear;
```

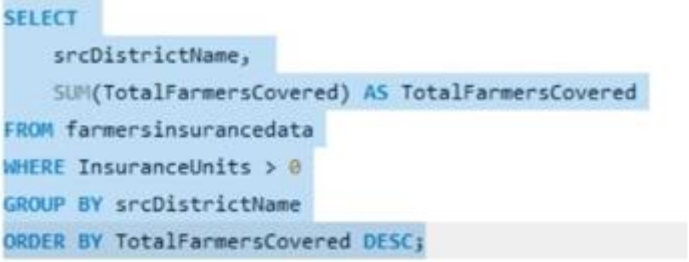
Output



```
srcYear  AvgInsuredLandArea
2018     38.235249986316866
2019     40.162162148120686
2020     48.35194718522074
2021     39.29111637346095
```

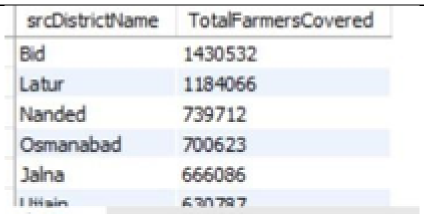
Q8. Calculate the total number of farmers covered (TotalFarmersCovered) for each district (srcDistrictName) where Insurance units is greater than 0.

Input



```
SELECT
    srcDistrictName,
    SUM(TotalFarmersCovered) AS TotalFarmersCovered
FROM farmersinsurancedata
WHERE InsuranceUnits > 0
GROUP BY srcDistrictName
ORDER BY TotalFarmersCovered DESC;
```

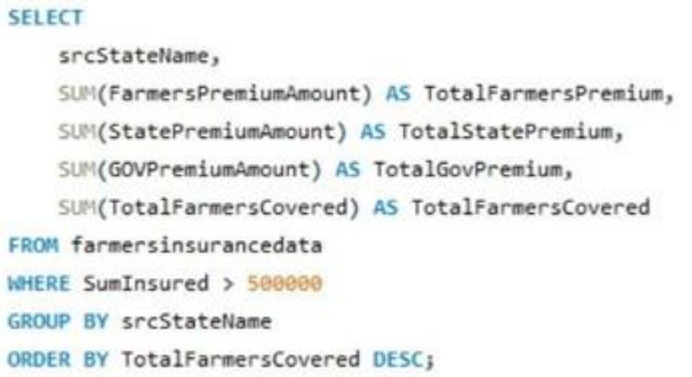
Output



```
srcDistrictName   TotalFarmersCovered
Bid               1430532
Latur             1184066
Nanded            739712
Osmanabad         700623
Jalna             666086
Ujjain            630787
```

Q9. For each state (srcStateName), calculate the total premium amounts (FarmersPremiumAmount, StatePremiumAmount, GOVPremiumAmount) and the total number of farmers covered (TotalFarmersCovered). Only include records where the sum insured (SumInsured) is greater than 500,000 (remember to check for scaling).

Input

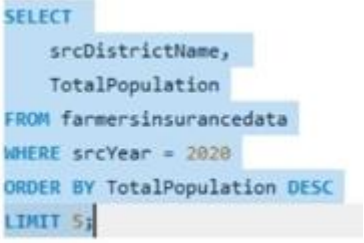


```
SELECT
    srcStateName,
    SUM(FarmersPremiumAmount) AS TotalFarmersPremium,
    SUM(StatePremiumAmount) AS TotalStatePremium,
    SUM(GOVPremiumAmount) AS TotalGovPremium,
    SUM(TotalFarmersCovered) AS TotalFarmersCovered
FROM farmersinsurancedata
WHERE SumInsured > 500000
GROUP BY srcStateName
ORDER BY TotalFarmersCovered DESC;
```

Output

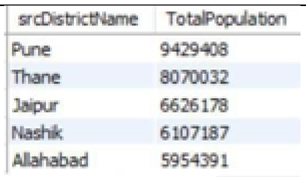
Q10. Retrieve the top 5 districts (srcDistrictName) with the highest TotalPopulation in the year 2020.

Input



```
SELECT
    srcDistrictName,
    TotalPopulation
FROM farmersinsurancedata
WHERE srcYear = 2020
ORDER BY TotalPopulation DESC
LIMIT 5;
```

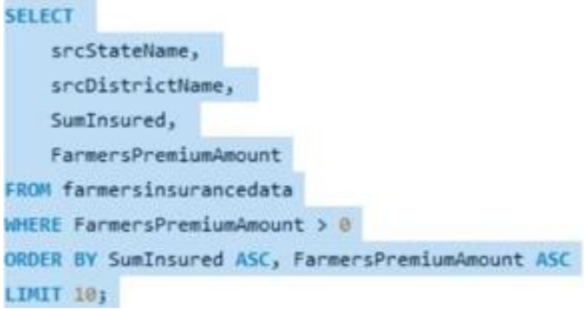
Output



srcDistrictName	TotalPopulation
Pune	9429408
Thane	8070032
Jaipur	6626178
Nashik	6107187
Allahabad	5954391

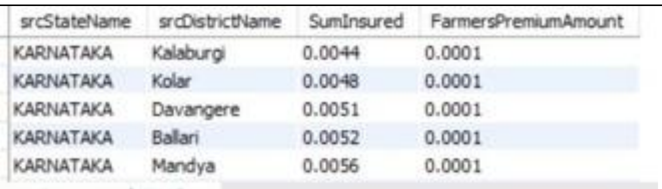
Q11. Retrieve the srcStateName, srcDistrictName, and SumInsured for the 10 districts with the lowest non-zero FarmersPremiumAmount, ordered by insured sum and then the FarmersPremiumAmount.

Input



```
SELECT
    srcStateName,
    srcDistrictName,
    SumInsured,
    FarmersPremiumAmount
FROM farmersinsurancedata
WHERE FarmersPremiumAmount > 0
ORDER BY SumInsured ASC, FarmersPremiumAmount ASC
LIMIT 10;
```

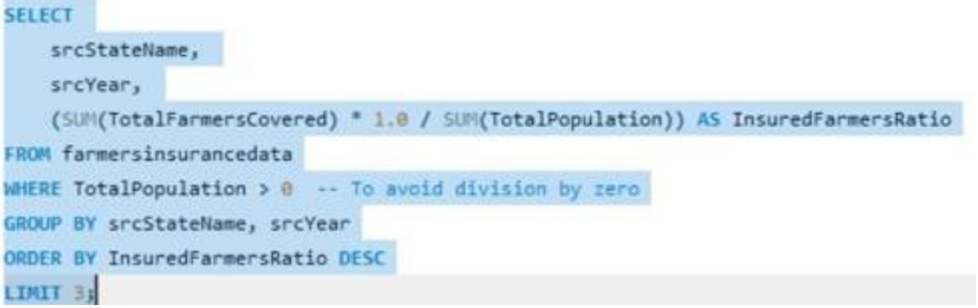
Output



srcStateName	srcDistrictName	SumInsured	FarmersPremiumAmount
KARNATAKA	Kalaburgi	0.0044	0.0001
KARNATAKA	Kolar	0.0048	0.0001
KARNATAKA	Davangere	0.0051	0.0001
KARNATAKA	Ballari	0.0052	0.0001
KARNATAKA	Mandya	0.0056	0.0001

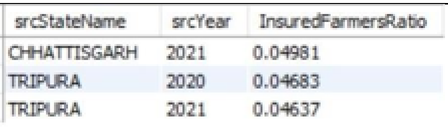
Q12. Retrieve the top 3 states (srcStateName) along with the year (srcYear) where the ratio of insured farmers (TotalFarmersCovered) to the total population (TotalPopulation) is highest. Sort the results by the ratio in descending order.

Input



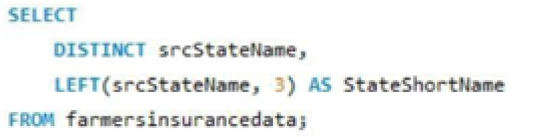
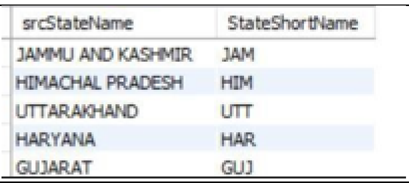
```
SELECT
    srcStateName,
    srcYear,
    (SUM(TotalFarmersCovered) * 1.0 / SUM(TotalPopulation)) AS InsuredFarmersRatio
FROM farmersinsurancedata
WHERE TotalPopulation > 0  -- To avoid division by zero
GROUP BY srcStateName, srcYear
ORDER BY InsuredFarmersRatio DESC
LIMIT 3;
```

Outpu
t

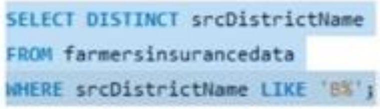
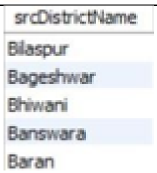


srcStateName	srcYear	InsuredFarmersRatio
CHHATTISGARH	2021	0.04981
TRIPURA	2020	0.04683
TRIPURA	2021	0.04637

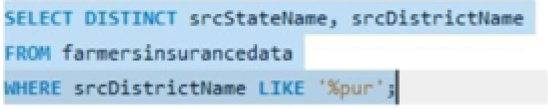
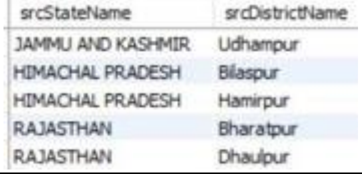
Q13. Create StateShortName by retrieving the first 3 characters of the srcStateName for each unique state.

<table>
<tr><td>Input</td><td>SELECT
    DISTINCT srcStateName,
    LEFT(srcStateName, 3) AS StateShortName
FROM farmersinsurancedata;</td></tr>
<tr><td>Output</td><td><table><tr><th>srcStateName</th><th>StateShortName</th></tr><tr><td>JAMMU AND KASHMIR</td><td>JAM</td></tr><tr><td>HIMACHAL PRADESH</td><td>HIM</td></tr><tr><td>UTTARAKHAND</td><td>UTT</td></tr><tr><td>HARYANA</td><td>HAR</td></tr><tr><td>GUJARAT</td><td>GUJ</td></tr></table></td></tr>
</table>

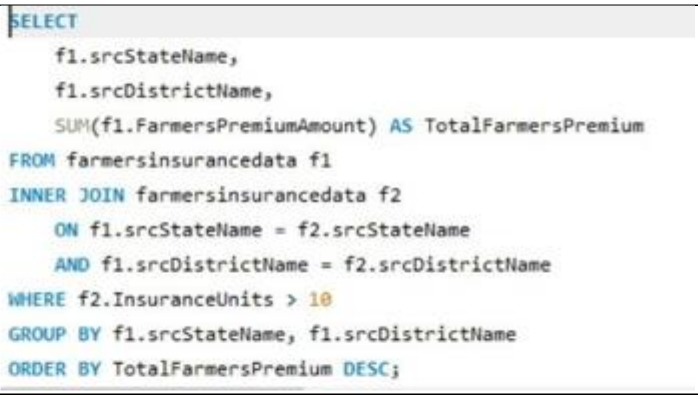
Q14. Retrieve the srcDistrictName where the district name starts with 'B'.

<table>
<tr><td>Input</td><td>SELECT DISTINCT srcDistrictName
FROM farmersinsurancedata
WHERE srcDistrictName LIKE 'B%';</td></tr>
<tr><td>Output</td><td><table><tr><th>srcDistrictName</th></tr><tr><td>Bilaspur</td></tr><tr><td>Bageshwar</td></tr><tr><td>Bhiwani</td></tr><tr><td>Banswara</td></tr><tr><td>Baran</td></tr></table></td></tr>
</table>

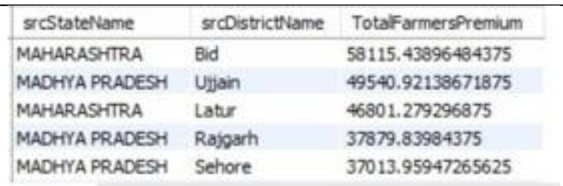
Q15. Retrieve the srcStateName and srcDistrictName where the district name contains the word 'pur' at the end.

<table>
<tr><td>Input</td><td>SELECT DISTINCT srcStateName, srcDistrictName
FROM farmersinsurancedata
WHERE srcDistrictName LIKE '%pur';</td></tr>
<tr><td>Output</td><td><table><tr><th>srcStateName</th><th>srcDistrictName</th></tr><tr><td>JAMMU AND KASHMIR</td><td>Udhampur</td></tr><tr><td>HIMACHAL PRADESH</td><td>Bilaspur</td></tr><tr><td>HIMACHAL PRADESH</td><td>Hamirpur</td></tr><tr><td>RAJASTHAN</td><td>Bharatpur</td></tr><tr><td>RAJASTHAN</td><td>Dhaulpur</td></tr></table></td></tr>
</table>

Q16. Perform an INNER JOIN between the srcStateName and srcDistrictName columns to retrieve the aggregated FarmersPremiumAmount for districts where the district's Insurance units for an individual year are greater than 10.

<table>
<tr><td>Input</td><td>SELECT
    f1.srcStateName,
    f1.srcDistrictName,
    SUM(f1.FarmersPremiumAmount) AS TotalFarmersPremium
FROM farmersinsurancedata f1
INNER JOIN farmersinsurancedata f2
    ON f1.srcStateName = f2.srcStateName
    AND f1.srcDistrictName = f2.srcDistrictName
WHERE f2.InsuranceUnits > 10
GROUP BY f1.srcStateName, f1.srcDistrictName
ORDER BY TotalFarmersPremium DESC;</td></tr>
</table>

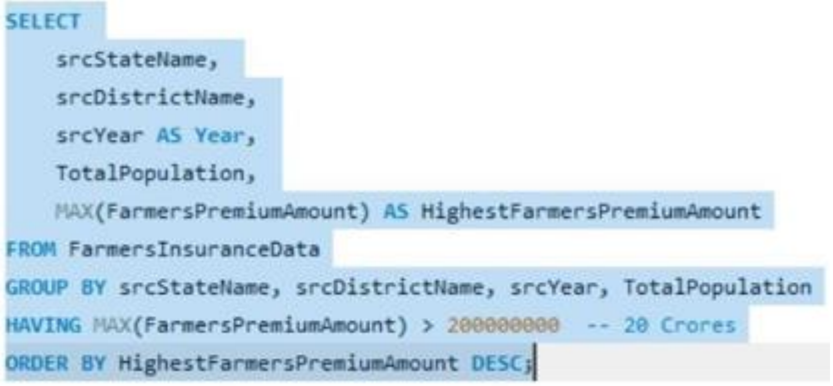
Output



srcStateName	srcDistrictName	TotalFarmersPremium
MAHARASHTRA	Bid	58115.43896484375
MADHYA PRADESH	Ujjain	49540.92138671875
MAHARASHTRA	Latur	46801.279296875
MADHYA PRADESH	Rajgarh	37879.83984375
MADHYA PRADESH	Sehore	37013.95947265625

Q17. Write a query that retrieves srcStateName, srcDistrictName, Year, TotalPopulation for each district and the the highest recorded FarmersPremiumAmount for that district over all available years Return only those districts where the highest FarmersPremiumAmount exceeds 20 crores.

Input



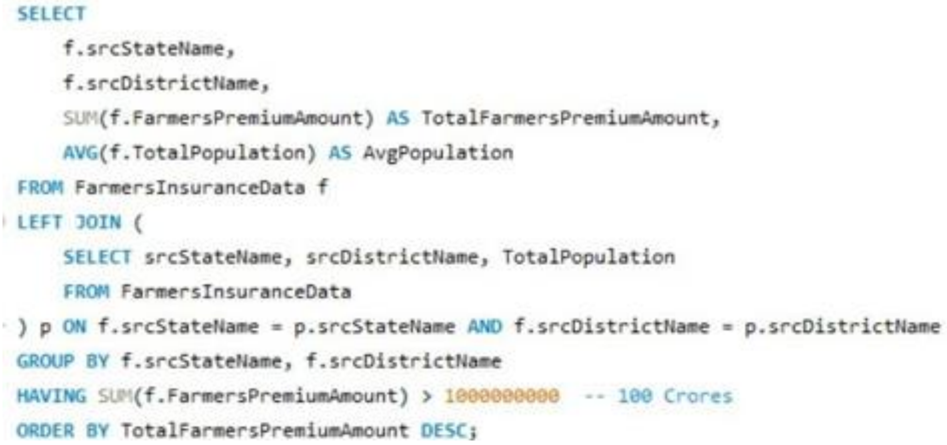
```
SELECT
    srcStateName,
    srcDistrictName,
    srcYear AS Year,
    TotalPopulation,
    MAX(FarmersPremiumAmount) AS HighestFarmersPremiumAmount
FROM FarmersInsuranceData
GROUP BY srcStateName, srcDistrictName, srcYear, TotalPopulation
HAVING MAX(FarmersPremiumAmount) > 200000000  -- 20 Crores
ORDER BY HighestFarmersPremiumAmount DESC;
```

Output

This query will return no results as dataset does not contain FarmersPremiumAmount values anywhere near 20 crores

Q18. Perform a LEFT JOIN to combine the total population statistics with the farmers’ data (TotalFarmersCovered, SumInsured) for each district and state. Return the total premium amount (FarmersPremiumAmount) and the average population count for each district aggregated over the years, where the total FarmersPremiumAmount is greater than 100 crores. Sort the results by total farmers' premium amount, highest first.

Input



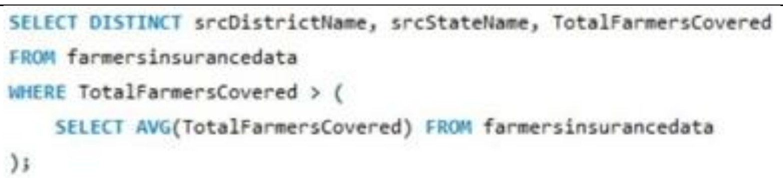
```
SELECT
    f.srcStateName,
    f.srcDistrictName,
    SUM(f.FarmersPremiumAmount) AS TotalFarmersPremiumAmount,
    AVG(f.TotalPopulation) AS AvgPopulation
FROM FarmersInsuranceData f
LEFT JOIN (
    SELECT srcStateName, srcDistrictName, TotalPopulation
    FROM FarmersInsuranceData
) p ON f.srcStateName = p.srcStateName AND f.srcDistrictName = p.srcDistrictName
GROUP BY f.srcStateName, f.srcDistrictName
HAVING SUM(f.FarmersPremiumAmount) > 1000000000  -- 100 Crores
ORDER BY TotalFarmersPremiumAmount DESC;
```

Output

This query will return no results as no districts meet the 100 crore threshold

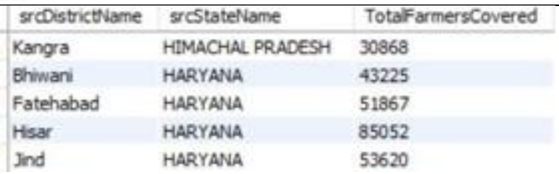
Q19. Write a query to find the districts (srcDistrictName) where the TotalFarmersCovered is greater than the average TotalFarmersCovered across all records.

Input



```
SELECT DISTINCT srcDistrictName, srcStateName, TotalFarmersCovered
FROM farmersinsurancedata
WHERE TotalFarmersCovered > (
    SELECT AVG(TotalFarmersCovered) FROM farmersinsurancedata
);
```

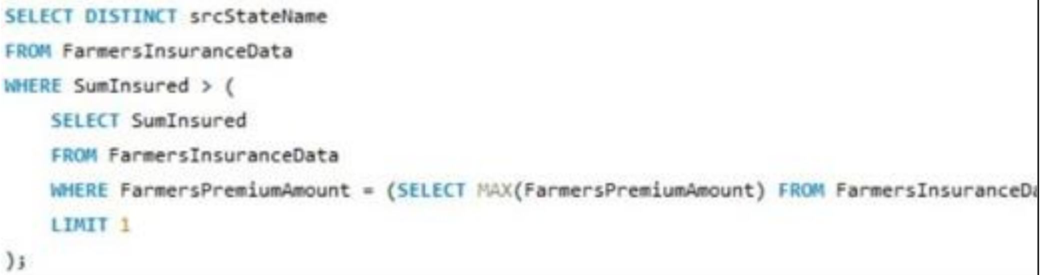
Output



```
srcDistrictName  srcStateName      TotalFarmersCovered
Kangra           HIMACHAL PRADESH  30868
Bhiwani          HARYANA           43225
Fatehabad        HARYANA           51867
Hisar            HARYANA           85052
Jind             HARYANA           53620
```

Q20. Write a query to find the srcStateName where the SumInsured is higher than the SumInsured of the district with the highest FarmersPremiumAmount.

Input



```
SELECT DISTINCT srcStateName
FROM FarmersInsuranceData
WHERE SumInsured > (
    SELECT SumInsured
    FROM FarmersInsuranceData
    WHERE FarmersPremiumAmount = (SELECT MAX(FarmersPremiumAmount) FROM FarmersInsuranceDa
    LIMIT 1
);
```

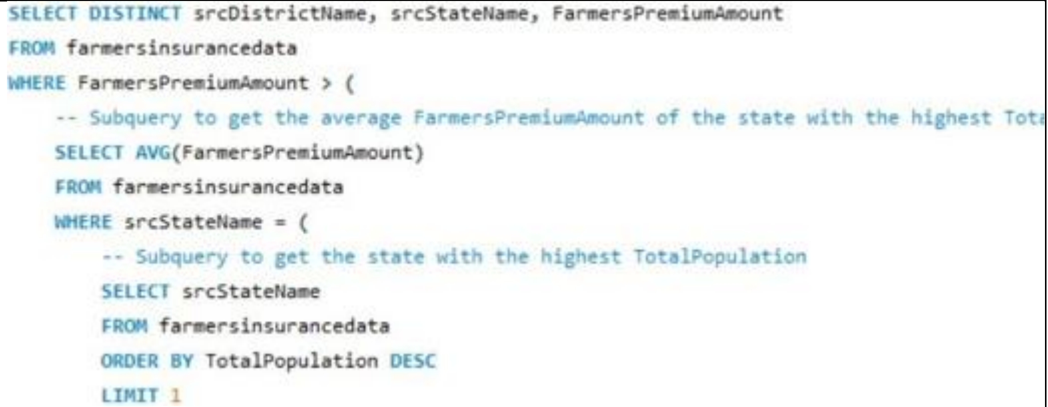
Output

From the dataset, the highest FarmersPremiumAmount is ₹7244.42 (for the district Bid, Maharashtra), and its SumInsured is ₹275,019.

Since the SumInsured threshold is very low, it's highly likely that many states do not exceed this value, which is why the query returned no results.

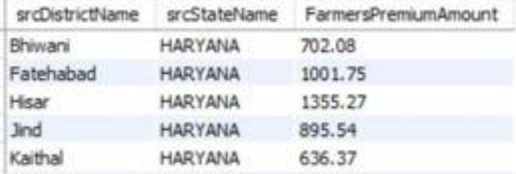
Q21. Write a query to find the srcDistrictName where the FarmersPremiumAmount is higher than the average FarmersPremiumAmount of the state that has the highest TotalPopulation.

Input



```
SELECT DISTINCT srcDistrictName, srcStateName, FarmersPremiumAmount
FROM farmersinsurancedata
WHERE FarmersPremiumAmount > (
    -- Subquery to get the average FarmersPremiumAmount of the state with the highest Tota
    SELECT AVG(FarmersPremiumAmount)
    FROM farmersinsurancedata
    WHERE srcStateName = (
        -- Subquery to get the state with the highest TotalPopulation
        SELECT srcStateName
        FROM farmersinsurancedata
        ORDER BY TotalPopulation DESC
        LIMIT 1
```

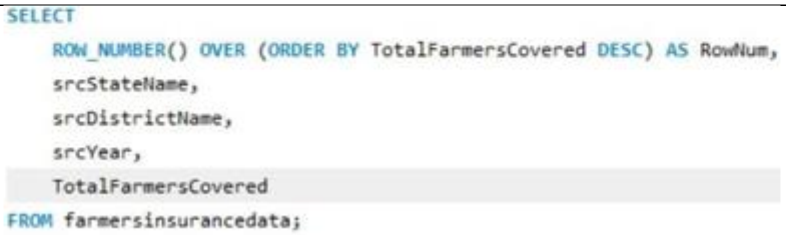
Output



```
srcDistrictName  srcStateName  FarmersPremiumAmount
Bhiwani          HARYANA       702.08
Fatehabad        HARYANA       1001.75
Hisar            HARYANA       1355.27
Jind             HARYANA       895.54
Kaithal          HARYANA       636.37
```

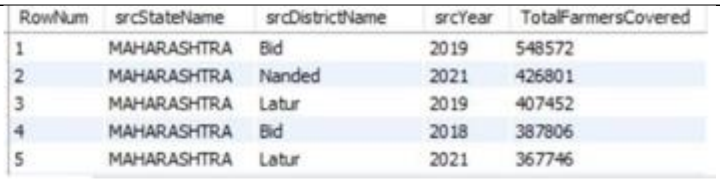
Q22. Use the ROW_NUMBER() function to assign a row number to each record in the dataset ordered by total farmers covered in descending order.

Input



```
SELECT
    ROW_NUMBER() OVER (ORDER BY TotalFarmersCovered DESC) AS RowNum,
    srcStateName,
    srcDistrictName,
    srcYear,
    TotalFarmersCovered
FROM farmersinsurancedata;
```

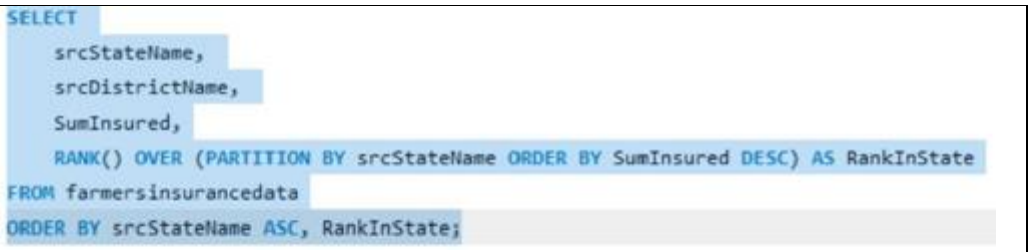
Output



RowNum	srcStateName	srcDistrictName	srcYear	TotalFarmersCovered
1	MAHARASHTRA	Bid	2019	548572
2	MAHARASHTRA	Nanded	2021	426801
3	MAHARASHTRA	Latur	2019	407452
4	MAHARASHTRA	Bid	2018	387806
5	MAHARASHTRA	Latur	2021	367746

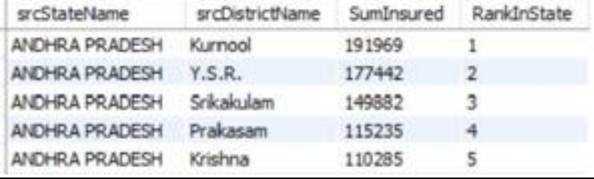
Q23. Use the RANK() function to rank the districts (srcDistrictName) based on the SumInsured (descending) and partition by alphabetical srcStateName.

Input



```
SELECT
    srcStateName,
    srcDistrictName,
    SumInsured,
    RANK() OVER (PARTITION BY srcStateName ORDER BY SumInsured DESC) AS RankInState
FROM farmersinsurancedata
ORDER BY srcStateName ASC, RankInState;
```

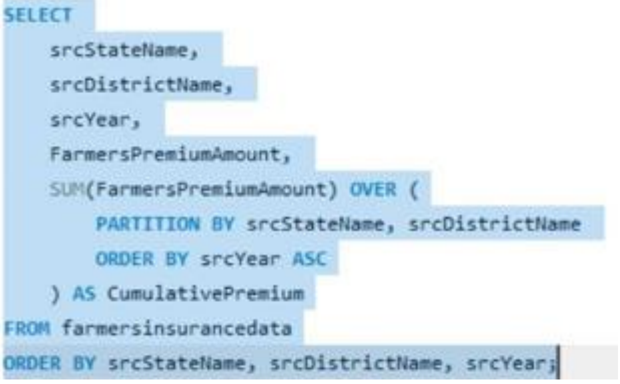
Outpu
t



srcStateName	srcDistrictName	SumInsured	RankInState
ANDHRA PRADESH	Kurnool	191969	1
ANDHRA PRADESH	Y.S.R.	177442	2
ANDHRA PRADESH	Srikakulam	149882	3
ANDHRA PRADESH	Prakasam	115235	4
ANDHRA PRADESH	Krishna	110285	5

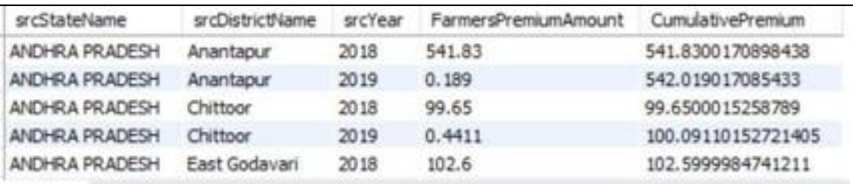
Q24. Use the SUM() window function to calculate a cumulative sum of FarmersPremiumAmount for each district (srcDistrictName), ordered ascending by the srcYear, partitioned by srcStateName.

Input



```
SELECT
    srcStateName,
    srcDistrictName,
    srcYear,
    FarmersPremiumAmount,
    SUM(FarmersPremiumAmount) OVER (
        PARTITION BY srcStateName, srcDistrictName
        ORDER BY srcYear ASC
    ) AS CumulativePremium
FROM farmersinsurancedata
ORDER BY srcStateName, srcDistrictName, srcYear;
```

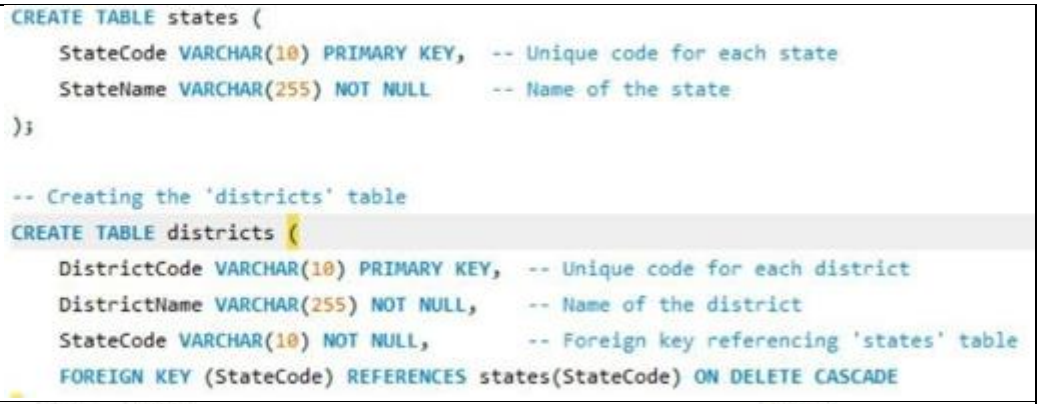
Output



srcStateName	srcDistrictName	srcYear	FarmersPremiumAmount	CumulativePremium
ANDHRA PRADESH	Anantapur	2018	541.83	541.8300170898438
ANDHRA PRADESH	Anantapur	2019	0.189	542.019017085433
ANDHRA PRADESH	Chittoor	2018	99.65	99.6500015258789
ANDHRA PRADESH	Chittoor	2019	0.4411	100.09110152721405
ANDHRA PRADESH	East Godavari	2018	102.6	102.5999984741211

Q25. Create a table 'districts' with DistrictCode as the primary key and columns for DistrictName and StateCode. Create another table 'states' with StateCode as primary key and column for StateName.

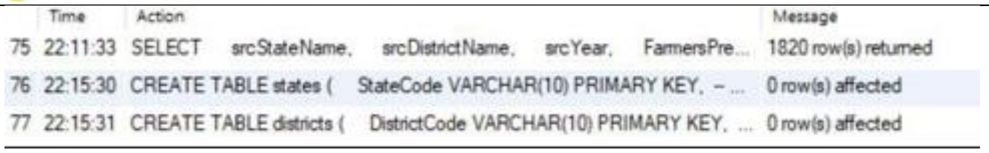
Input



```
CREATE TABLE states (
    StateCode VARCHAR(10) PRIMARY KEY,  -- Unique code for each state
    StateName VARCHAR(255) NOT NULL     -- Name of the state
);

-- Creating the 'districts' table
CREATE TABLE districts (
    DistrictCode VARCHAR(10) PRIMARY KEY,  -- Unique code for each district
    DistrictName VARCHAR(255) NOT NULL,    -- Name of the district
    StateCode VARCHAR(10) NOT NULL,        -- Foreign key referencing 'states' table
    FOREIGN KEY (StateCode) REFERENCES states(StateCode) ON DELETE CASCADE
```

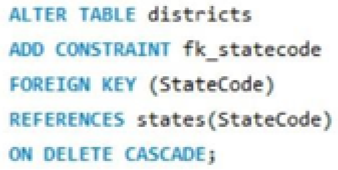
Outp
ut



```
    Time      Action                                                                  Message
75  22:11:33  SELECT   srcStateName,   srcDistrictName,   srcYear,   FarmersPre...     1820 row(s) returned
76  22:15:30  CREATE TABLE states (   StateCode VARCHAR(10) PRIMARY KEY, -- ...       0 row(s) affected
77  22:15:31  CREATE TABLE districts (   DistrictCode VARCHAR(10) PRIMARY KEY, ...    0 row(s) affected
```

Q26. Add a foreign key constraint to the districts table that references the StateCode column from a states table.

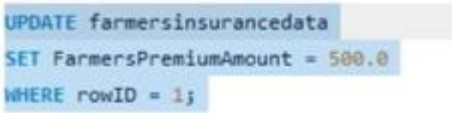
Input



```
ALTER TABLE districts
ADD CONSTRAINT fk_statecode
FOREIGN KEY (StateCode)
REFERENCES states(StateCode)
ON DELETE CASCADE;
```

Q27. Update the FarmersPremiumAmount to 500.0 for the record where rowID is 1.

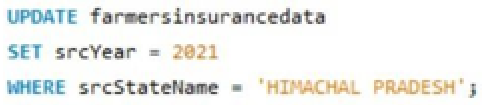
Input



```
UPDATE farmersinsurancedata
SET FarmersPremiumAmount = 500.0
WHERE rowID = 1;
```

Q28. Update the Year to '2021' for all records where srcStateName is 'HIMACHAL PRADESH'.

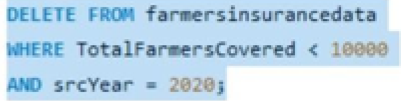
Input



```
UPDATE farmersinsurancedata
SET srcYear = 2021
WHERE srcStateName = 'HIMACHAL PRADESH';
```

Q29. Delete all records where the TotalFarmersCovered is less than 10000 and Year is 2020.

Input



```
DELETE FROM farmersinsurancedata
WHERE TotalFarmersCovered < 10000
AND srcYear = 2020;
```