Bharat Herald: Ad-Hoc Request Report

Submitted by: Peter Pandey (Bala Marimuthu)

Date: September 28, 2025

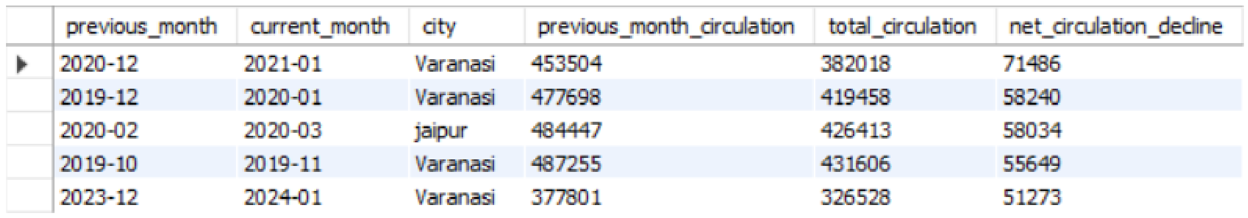
Request 1: Monthly Circulation Drop Check

Generate a report showing the top 3 months (2019–2024) where any city recorded the sharpest month-over-month decline in net_circulation.

SQL Query:

```
WITH date_converted_sales AS (
   SELECT
      t1.Net_circulation,
      t2.city,
      CASE
         WHEN TRIM(t1.sale_month) LIKE '%/%' THEN
SUBSTRING_INDEX(TRIM(t1.sale_month), '/', 1)
         WHEN TRIM(t1.sale_month) LIKE '%-%' THEN CONCAT('20',
SUBSTRING_INDEX(TRIM(t1.sale_month), '-', -1))
      END AS sales_year,
      CASE
         WHEN TRIM(t1.sale_month) LIKE '%/%' THEN
SUBSTRING_INDEX(TRIM(t1.sale_month), '/', -1)
         WHEN TRIM(t1.sale_month) LIKE '%-%' THEN
            CASE LOWER(SUBSTRING_INDEX(TRIM(t1.sale_month), '-', 1))   -- CONVERT INTO
LOWERCASE
               WHEN 'jan' THEN 1  WHEN 'feb' THEN 2  WHEN 'mar' THEN 3
               WHEN 'apr' THEN 4  WHEN 'may' THEN 5  WHEN 'jun' THEN 6
               WHEN 'jul' THEN 7  WHEN 'aug' THEN 8  WHEN 'sep' THEN 9
               WHEN 'oct' THEN 10 WHEN 'nov' THEN 11 WHEN 'dec' THEN 12
            END
      END AS sales_month
   FROM
      fact_print_sales AS t1
   JOIN
      dim_city AS t2 ON t1.city_id = t2.city_id
),
monthly_circulation AS (
  SELECT
    sales_year,
    sales_month,
    city,
    SUM(Net_circulation) AS total_circulation
  FROM
    date_converted_sales
  WHERE
    sales_year BETWEEN 2019 AND 2024
  GROUP BY
    sales_year, sales_month, city
),
mom_change AS (
  SELECT
    sales_year,
    sales_month,
    city,
    total_circulation,
    LAG(total_circulation, 1, 0) OVER (PARTITION BY city ORDER BY sales_year,
sales_month) AS previous_month_circulation
  FROM
    monthly_circulation
)
SELECT
  -- Added a 'previous_month' column for clarity
  DATE_FORMAT((MAKEDATE(sales_year, 1) + INTERVAL sales_month - 1 MONTH) -
INTERVAL 1 MONTH, '%Y-%m') AS previous_month,
  -- Renamed the original 'month' column
  DATE_FORMAT(MAKEDATE(sales_year, 1) + INTERVAL sales_month - 1 MONTH, '%Y-%m')
AS current_month,
  city,
  previous_month_circulation,
  total_circulation,
  (previous_month_circulation - total_circulation) AS net_circulation_decline
FROM
  mom_change
WHERE
  previous_month_circulation > total_circulation
ORDER BY
  net_circulation_decline DESC
LIMIT 5;
```

Result:



previous_month	current_month	city	previous_month_circulation	total_circulation	net_circulation_decline
2020-12	2021-01	Varanasi	453504	382018	71486
2019-12	2020-01	Varanasi	477698	419458	58240
2020-02	2020-03	jaipur	484447	426413	58034
2019-10	2019-11	Varanasi	487255	431606	55649
2023-12	2024-01	Varanasi	377801	326528	51273

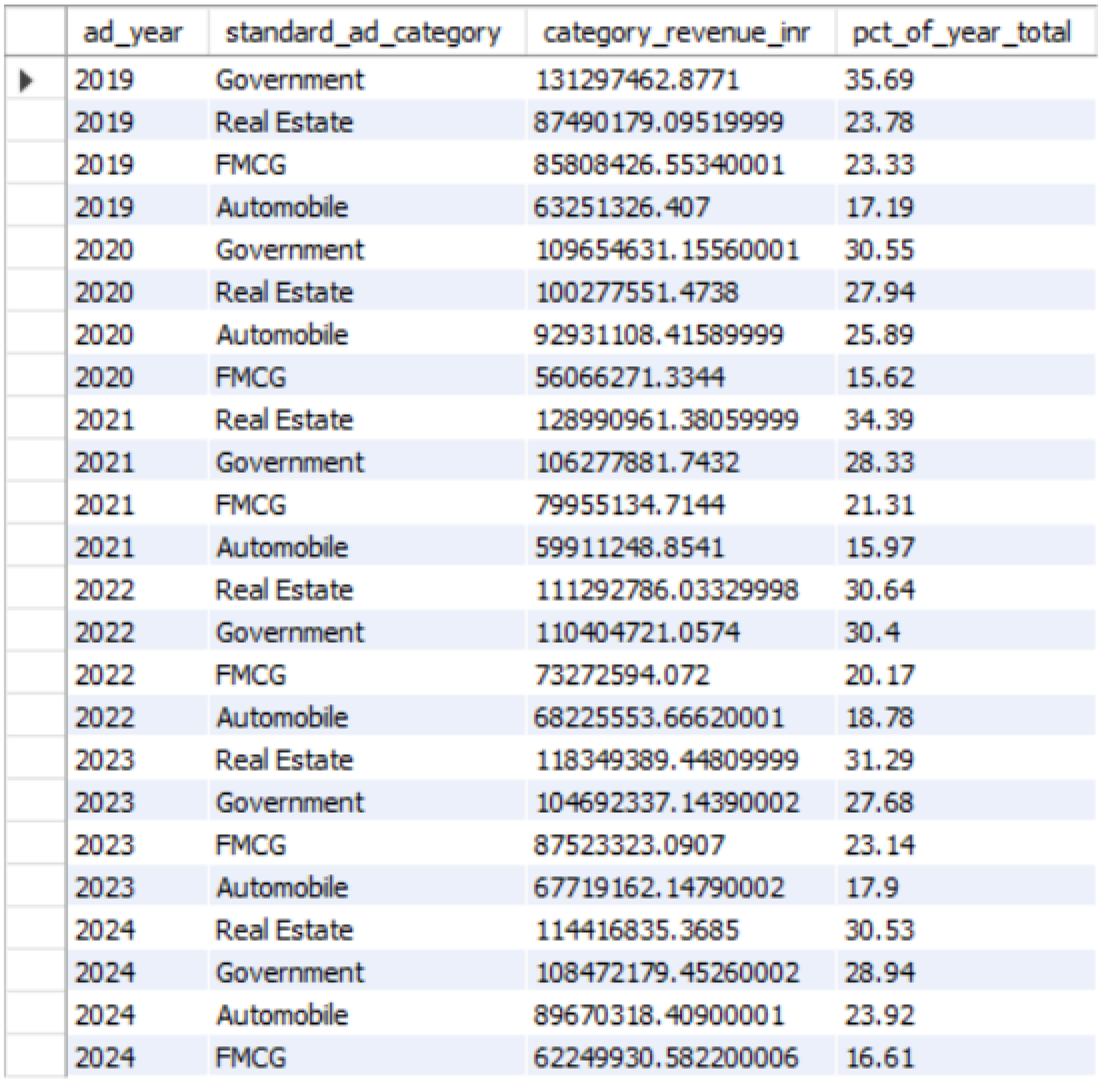
Request 2: Ad Category Contribution

Identify ad categories that contributed > 50% of total yearly ad revenue.

SQL Query:

```
with clean_yearly_revenue as(
select
       case
              when locate('-', t1.time_quarter)=5 then left (t1.time_quarter,4)
    when locate('-',t1.time_quarter)=3 then right (t1.time_quarter,4)
    else right(t1.time_quarter,4)
       end as ad_year,
  t2.standard_ad_category,
  sum(
              case
                     when t1.currency='USD'  then t1.ad_revenue *88.14
      when t1.currency='EUR'  then t1.ad_revenue * 103.07
      when t1.currency in ('INR' ,'IN RUPEES') then t1.ad_revenue
      else t1.ad_revenue
              end
) as category_revenue_inr
from
       fact_ad_revenue as t1
join
       dim_ad_category as t2 on t1.ad_category = t2.ad_category_id
  group by
              ad_year , t2.standard_ad_category
)
 select
       ad_year,
  standard_ad_category,
  category_revenue_inr,
   round(
                    (category_revenue_inr/ sum(category_revenue_inr) over (partition by
ad_year)) *100,
       2
           ) as pct_of_year_total
from
           clean_yearly_revenue
order by
           ad_year, pct_of_year_total desc;
```

Result:



ad_year	standard_ad_category	category_revenue_inr	pct_of_year_total
2019	Government	131297462.8771	35.69
2019	Real Estate	87490179.09519999	23.78
2019	FMCG	85808426.55340001	23.33
2019	Automobile	63251326.407	17.19
2020	Government	109654631.15560001	30.55
2020	Real Estate	100277551.4738	27.94
2020	Automobile	92931108.41589999	25.89
2020	FMCG	56066271.3344	15.62
2021	Real Estate	128990961.38059999	34.39
2021	Government	106277881.7432	28.33
2021	FMCG	79955134.7144	21.31
2021	Automobile	59911248.8541	15.97
2022	Real Estate	111292786.03329998	30.64
2022	Government	110404721.0574	30.4
2022	FMCG	73272594.072	20.17
2022	Automobile	68225553.66620001	18.78
2023	Real Estate	118349389.44809999	31.29
2023	Government	104692337.14390002	27.68
2023	FMCG	87523323.0907	23.14
2023	Automobile	67719162.14790002	17.9
2024	Real Estate	114416835.3685	30.53
2024	Government	108472179.45260002	28.94
2024	Automobile	89670318.40900001	23.92
2024	FMCG	62249930.582200006	16.61

Request 3: Print Efficiency Ranking

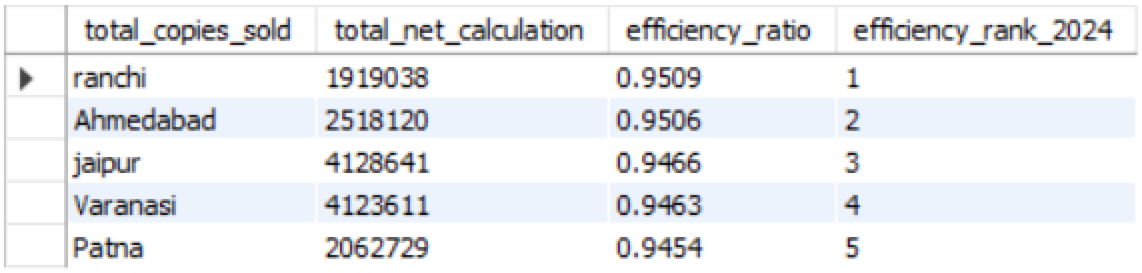
For 2024, rank cities by print efficiency = net_circulation / copies_printed. Return top 5.

SQL Query:

```
 with city_efficinecy_2024 as(
select
          t2.city ,
   sum(t1.Net_Circulation) as total_net_calculation,
   sum(t1.Copies_Sold) as total_copies_sold,
   -- calcualte print efficiency
   (sum(t1.Net_Circulation)/sum(t1.Copies_Sold)) as efficiency_ratio
   from
   fact_print_sales as t1
   join
   dim_city as t2 on t1.City_ID =t2.city_id
   where
   case
                    when trim(t1.sale_month) like "%/%" then
SUBSTRING_INDEX(trim(t1.sale_month),'/',1)
      when trim(t1.sale_month) like "%-%" then
concat(20,SUBSTRING_INDEX(trim(t1.sale_month),'-',-1))
          end = '2024'

group by
          t2.city
)
select
city
total_copies_sold,
total_net_calculation,
efficiency_ratio,
-- addrank based effiecienccy ratio
rank() OVER(ORDER BY efficiency_ratio desc) as efficiency_rank_2024
from
city_efficinecy_2024
order by
efficiency_rank_2024
limit 5;
```

Result:



	total_copies_sold	total_net_calculation	efficiency_ratio	efficiency_rank_2024
▶	ranchi	1919038	0.9509	1
	Ahmedabad	2518120	0.9506	2
	jaipur	4128641	0.9466	3
	Varanasi	4123611	0.9463	4
	Patna	2062729	0.9454	5

Request 4: Internet Penetration Change

For each city, compute the change in internet penetration from Q1-2021 to Q4-2021 and identify the city with the highest improvement.

SQL Query:

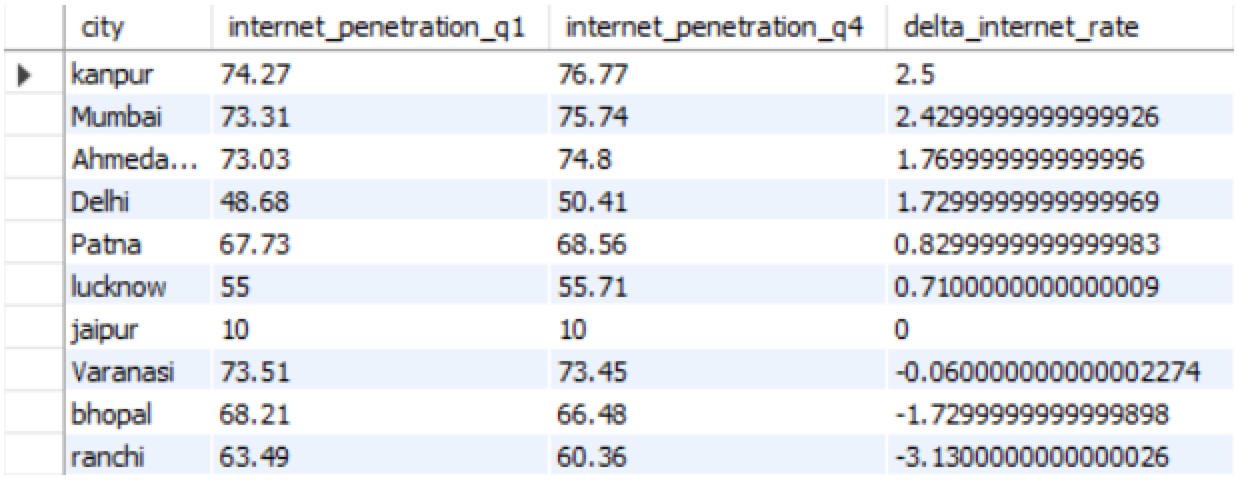
```
with q1_rates as(
select
       city_id ,
  internet_penetration as internet_penetration_q1
  from
  fact_city_readiness
  where
  time_quarter ='2021-Q1'
),
q4_rates as (
       select
  city_id,
    internet_penetration as internet_penetration_q4
       from
  fact_city_readiness
  where
  time_quarter ='2021-Q4'
)
select
```

```
        c.city,
    q1.internet_penetration_q1,
    q4.internet_penetration_q4,
    (q4.internet_penetration_q4 -q1.internet_penetration_q1) as delta_internet_rate

from
        q1_rates as q1
join
        q4_rates as q4 on q1.city_id =q4.city_id

join
        dim_city as c on q1.city_id =c.city_id
order by
delta_internet_rate desc
limit 3;
```

Result:



city	internet_penetration_q1	internet_penetration_q4	delta_internet_rate
kanpur	74.27	76.77	2.5
Mumbai	73.31	75.74	2.4299999999999926
Ahmeda...	73.03	74.8	1.769999999999996
Delhi	48.68	50.41	1.7299999999999969
Patna	67.73	68.56	0.8299999999999983
lucknow	55	55.71	0.7100000000000009
jaipur	10	10	0
Varanasi	73.51	73.45	-0.060000000000002274
bhopal	68.21	66.48	-1.7299999999999898
ranchi	63.49	60.36	-3.1300000000000026

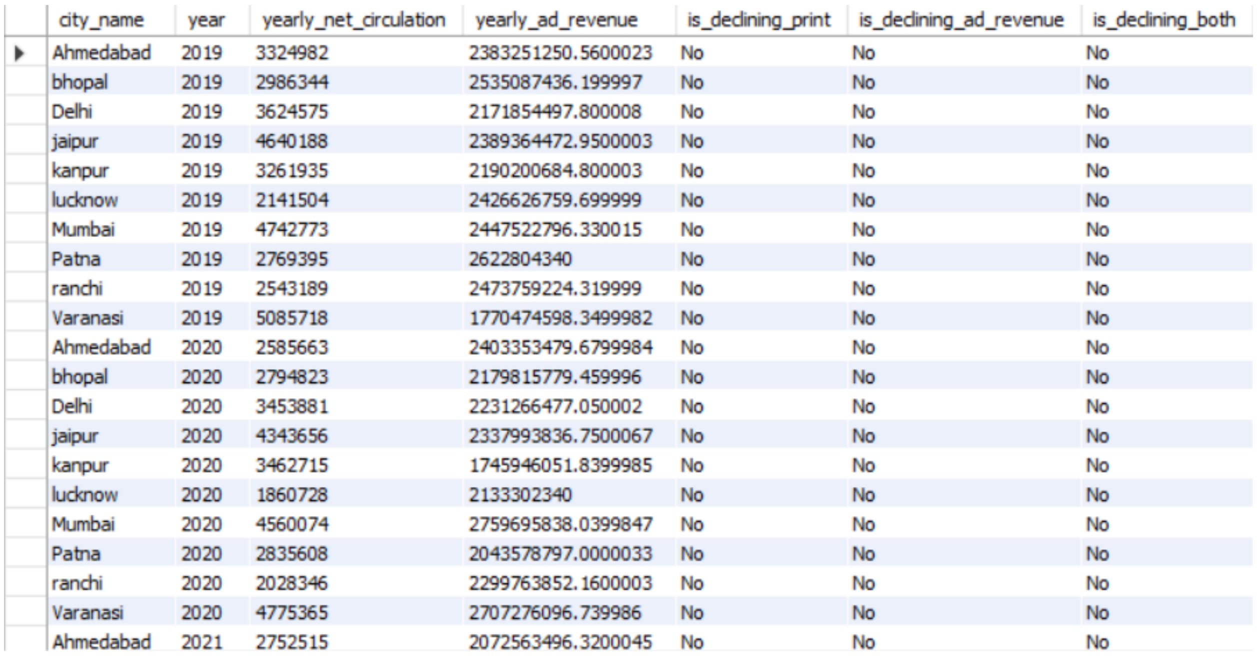
Request 5: Strictly Decreasing Trends

Find cities where both net_circulation and ad_revenue decreased every year from 2019 through 2024 (strictly decreasing sequences).

SQL Query:

```
-- Step 1: Aggregate yearly circulation for each city
WITH yearly_circulation AS (
  SELECT
    t1.City_ID,
    CASE
      WHEN TRIM(t1.sale_month) LIKE '%/%' THEN
SUBSTRING_INDEX(TRIM(t1.sale_month), '/', 1)
      WHEN TRIM(t1.sale_month) LIKE '%-%' THEN CONCAT('20',
SUBSTRING_INDEX(TRIM(t1.sale_month), '-', -1))
    END AS metric_year,
    SUM(t1.Net_Circulation) AS yearly_net_circulation
  FROM
    fact_print_sales AS t1
  GROUP BY
    t1.City_ID, metric_year
),
-- Step 2: Aggregate yearly ad revenue for each city by linking through edition_id
yearly_revenue AS (
  SELECT
    fps.City_ID,
    CASE
      WHEN LOCATE('-', far.time_quarter) = 5 THEN LEFT(far.time_quarter, 4)
      WHEN LOCATE('-', far.time_quarter) = 3 THEN RIGHT(far.time_quarter, 4)
      ELSE RIGHT(far.time_quarter, 4)
    END AS metric_year,
    SUM(
      CASE
        WHEN far.currency = 'USD' THEN far.ad_revenue * 83.0
        WHEN far.currency = 'EUR' THEN far.ad_revenue * 90.0
        WHEN far.currency IN ('INR', 'IN RUPEES') THEN far.ad_revenue
        ELSE far.ad_revenue
      END
    ) AS yearly_ad_revenue
  FROM fact_ad_revenue AS far
  -- Join through fact_print_sales to get the City_ID
  JOIN fact_print_sales AS fps ON far.edition_id = fps.edition_id
  GROUP BY
    fps.City_ID, metric_year
),
-- Step 3: Combine the two aggregated datasets
yearly_aggregated_data AS (
  SELECT
    yc.City_ID,
    yc.metric_year,
        yc.yearly_net_circulation,
        yr.yearly_ad_revenue
    FROM yearly_circulation AS yc
    JOIN yearly_revenue AS yr ON yc.City_ID = yr.City_ID AND yc.metric_year =
yr.metric_year
    WHERE
        yc.metric_year BETWEEN '2019' AND '2024'
),
-- Step 4: Compare each year to the previous year using LAG()
yearly_comparison AS (
    SELECT
        City_ID,
        metric_year,
        yearly_net_circulation,
        yearly_ad_revenue,
        LAG(yearly_net_circulation, 1) OVER (PARTITION BY City_ID ORDER BY metric_year)
AS prev_year_circ,
        LAG(yearly_ad_revenue, 1) OVER (PARTITION BY City_ID ORDER BY metric_year) AS
prev_year_rev
    FROM yearly_aggregated_data
),
-- Step 5: Check the trend for each city
trend_check AS (
    SELECT
        City_ID,
        COUNT(DISTINCT metric_year) AS total_years,
        SUM(CASE WHEN yearly_net_circulation < prev_year_circ THEN 1 ELSE 0 END) AS
circ_decrease_count,
        SUM(CASE WHEN yearly_ad_revenue < prev_year_rev THEN 1 ELSE 0 END) AS
rev_decrease_count
    FROM yearly_comparison
    WHERE metric_year > '2019'
    GROUP BY City_ID
)
-- Step 6: Join the yearly data with the trend flags for the final report
SELECT
    c.city AS city_name,
    y.metric_year AS year,
    y.yearly_net_circulation,
    y.yearly_ad_revenue,
    CASE
        WHEN tc.total_years = 6 AND tc.circ_decrease_count = 5 THEN 'Yes'
        ELSE 'No'
    END AS is_declining_print,
    CASE
        WHEN tc.total_years = 6 AND tc.rev_decrease_count = 5 THEN 'Yes'
        ELSE 'No'
    END AS is_declining_ad_revenue,
    CASE
        WHEN tc.total_years = 6 AND tc.circ_decrease_count = 5 AND tc.rev_decrease_count
= 5 THEN 'Yes'
        ELSE 'No'
    END AS is_declining_both
FROM
    yearly_aggregated_data AS y
JOIN
    dim_city AS c ON y.City_ID = c.city_id
LEFT JOIN
    trend_check AS tc ON y.City_ID = tc.City_ID
ORDER BY
    city_name, year;
```

Result:



city_name	year	yearly_net_circulation	yearly_ad_revenue	is_declining_print	is_declining_ad_revenue	is_declining_both
Ahmedabad	2019	3324982	2383251250.5600023	No	No	No
bhopal	2019	2986344	2535087436.199997	No	No	No
Delhi	2019	3624575	2171854497.800008	No	No	No
jaipur	2019	4640188	2389364472.9500003	No	No	No
kanpur	2019	3261935	2190200684.800003	No	No	No
lucknow	2019	2141504	2426626759.699999	No	No	No
Mumbai	2019	4742773	2447522796.330015	No	No	No
Patna	2019	2769395	2622804340	No	No	No
ranchi	2019	2543189	2473759224.319999	No	No	No
Varanasi	2019	5085718	1770474598.3499982	No	No	No
Ahmedabad	2020	2585663	2403353479.6799984	No	No	No
bhopal	2020	2794823	2179815779.459996	No	No	No
Delhi	2020	3453881	2231266477.050002	No	No	No
jaipur	2020	4343656	2337993836.7500067	No	No	No
kanpur	2020	3462715	1745946051.8399985	No	No	No
lucknow	2020	1860728	2133302340	No	No	No
Mumbai	2020	4560074	2759695838.0399847	No	No	No
Patna	2020	2835608	2043578797.0000033	No	No	No
ranchi	2020	2028346	2299763852.1600003	No	No	No
Varanasi	2020	4775365	2707276096.739986	No	No	No
Ahmedabad	2021	2752515	2072563496.3200045	No	No	No

Note: The full result set contains over 40 rows, providing a detailed year-by-year report for all cities. The full data is available upon request.

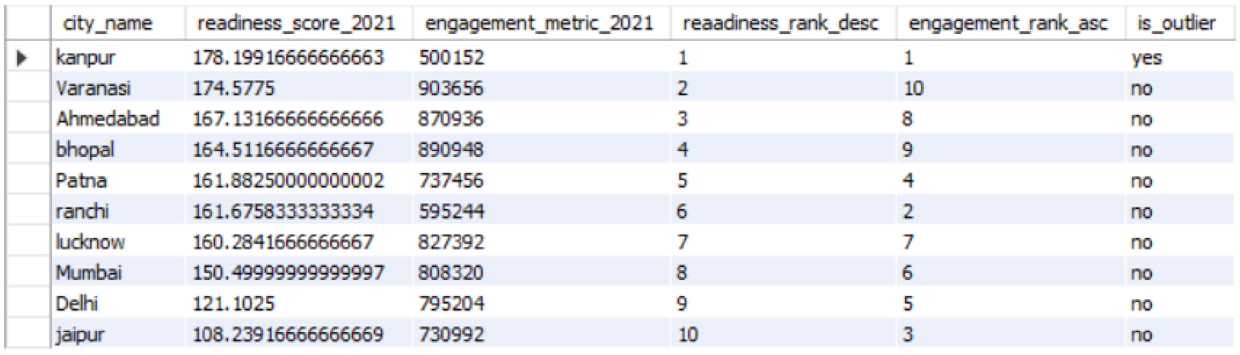
Request 6: Digital Readiness Outlier

In 2021, identify the city with the highest digital readiness score but among the bottom 3 in digital pilot engagement.

SQL Query:

```
with city_scores_2021 as(
select
	cr.city_id,
   avg(cr.smartphone_penetration +cr.internet_penetration+cr.literacy_rate/3) as
readiness_score_2021,
   sum(dp.users_reached+dp.downloads_or_accesses) as engagement_metric_2021
from
	fact_city_readiness as cr
join fact_digital_pilot as dp on cr.city_id =dp.city_id
where
	left (trim(cr.time_quarter),4)='2021' and
   left(trim(dp.launch_month),4)='2021'
group by
	cr.city_id
),
city_ranks as (
	select
		city_id,
      readiness_score_2021,
      engagement_metric_2021,
      rank() over (order by readiness_score_2021 desc) as reaadiness_rank_desc,
      rank() over (order by engagement_metric_2021 asc) as engagement_rank_asc
	from
		city_scores_2021
)
select
	c.city as city_name,
   cr.readiness_score_2021,
   cr.engagement_metric_2021,
   cr.reaadiness_rank_desc,
   cr.engagement_rank_asc,
   case
                     when cr.city_id=(
      SELECT city_id from city_ranks
      where engagement_rank_asc <= 3
      order by reaadiness_rank_desc asc
      limit 1
           ) then "yes"
   else "no"
   end as is_outlier
from
city_ranks as cr
join
dim_city as c on cr.city_id =c.city_id
order by
reaadiness_rank_desc;
```

Result:



city_name	readiness_score_2021	engagement_metric_2021	reaadiness_rank_desc	engagement_rank_asc	is_outlier
kanpur	178.19916666666663	500152	1	1	yes
Varanasi	174.5775	903656	2	10	no
Ahmedabad	167.13166666666666	870936	3	8	no
bhopal	164.5116666666667	890948	4	9	no
Patna	161.88250000000002	737456	5	4	no
ranchi	161.6758333333334	595244	6	2	no
lucknow	160.2841666666667	827392	7	7	no
Mumbai	150.49999999999997	808320	8	6	no
Delhi	121.1025	795204	9	5	no
jaipur	108.23916666666669	730992	10	3	no