

Cortex Analyst Semantics – Semantic Model Generator Prompts and Responses

Welcome to the Snowflake Semantic Model Generator! ❄️

Let's get started. What would you like to do?

🔗 Create a new semantic model

✎ Edit an existing semantic model

🤖 Start with partner semantic model

1. Prompt

Calculate total, min, max and avg revenue per month, providing the dates on which min and max sales occurred

Answer 1:

```
WITH __player_activity_table AS (
  SELECT
    user_id,
    login_date,
    ad_revenue,
    iap_revenue
  FROM cortex_analyst_semantics.semantic_model_generator.player_activity_table
), __player_data_table AS (
  SELECT
    user_id
  FROM cortex_analyst_semantics.semantic_model_generator.player_data_table
), monthly_revenue AS (
  SELECT
    DATE_TRUNC('MONTH', pa.login_date) AS month,
    SUM(pa.ad_revenue + pa.iap_revenue) AS total_revenue,
    MIN(pa.ad_revenue + pa.iap_revenue) AS min_revenue,
    MAX(pa.ad_revenue + pa.iap_revenue) AS max_revenue,
    AVG(pa.ad_revenue + pa.iap_revenue) AS avg_revenue
  FROM __player_activity_table AS pa
  INNER JOIN __player_data_table AS pd
    ON pa.user_id = pd.user_id
  GROUP BY
    DATE_TRUNC('MONTH', pa.login_date)
), min_max_dates AS (
  SELECT
    month,
    MIN(
      CASE
        WHEN (
          pa.ad_revenue + pa.iap_revenue
        ) = mr.min_revenue
```

```

        THEN pa.login_date
    END
) AS min_revenue_date,
MAX(
    CASE
        WHEN (
            pa.ad_revenue + pa.iap_revenue
        ) = mr.max_revenue
        THEN pa.login_date
    END
) AS max_revenue_date
FROM monthly_revenue AS mr
INNER JOIN __player_activity_table AS pa
    ON DATE_TRUNC('MONTH', pa.login_date) = mr.month
GROUP BY
    month,
    mr.min_revenue,
    mr.max_revenue
)
SELECT
    mr.month,
    mr.total_revenue,
    mr.min_revenue,
    mr.max_revenue,
    mr.avg_revenue,
    mmd.min_revenue_date,
    mmd.max_revenue_date
FROM monthly_revenue AS mr
LEFT JOIN min_max_dates AS mmd
    ON mr.month = mmd.month
ORDER BY
    mr.month DESC
-- Generated by Cortex Analyst (learnt in two trials)
;

```

MONTH	TOTAL_REVENUE	MIN_REVENUE	MAX_REVENUE	AVG_REVENUE	MIN_REVENUE_DATE	MAX_REVENUE_DATE
2021-12-01	11,786	0	753	1.0707	2021-12-01	2021-12-27
2021-11-01	10,523	0	785	1.0916	2021-11-01	2021-11-03
2021-10-01	10,497	0	749	0.9758	2021-10-01	2021-10-06
2021-09-01	17,362	0	753	1.1963	2021-09-01	2021-09-24
2021-08-01	10,993	0	752	0.8326	2021-08-01	2021-08-14
2021-07-01	14,193	0	721	0.9604	2021-07-01	2021-07-31
2021-06-01	15,255	0	780	1.1187	2021-06-01	2021-06-30
2021-05-01	18,393	0	784	1.3159	2021-05-01	2021-05-03
2021-04-01	13,351	0	789	0.9033	2021-04-01	2021-04-18
2021-03-01	11,140	0	775	0.6356	2021-03-01	2021-03-14
2021-02-01	12,722	0	2,284	0.7471	2021-02-01	2021-02-21

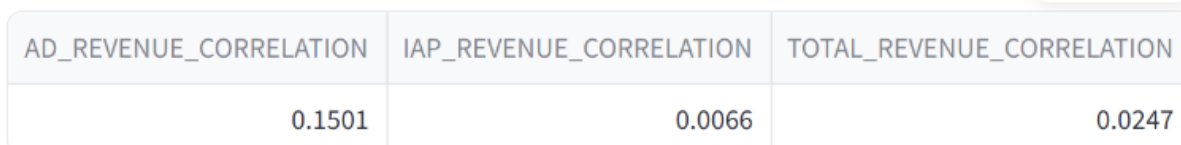
2. Prompt

Does playing longer generate bigger income? What kind?

Answer:

```
WITH __player_activity_table AS (  
  SELECT  
    user_id,  
    total_time_sec,  
    ad_revenue,  
    iap_revenue  
  FROM cortex_analyst_semantics.semantic_model_generator.player_activity_table  
) , __player_data_table AS (  
  SELECT  
    user_id  
  FROM cortex_analyst_semantics.semantic_model_generator.player_data_table  
) , player_activity AS (  
  SELECT  
    pa.user_id,  
    pa.total_time_sec,  
    pa.ad_revenue,  
    pa.iap_revenue  
  FROM __player_activity_table AS pa  
  INNER JOIN __player_data_table AS pd  
    ON pa.user_id = pd.user_id  
) , income_analysis AS (  
  SELECT  
    user_id,  
    total_time_sec,  
    ad_revenue,  
    iap_revenue,  
    ad_revenue + iap_revenue AS total_revenue  
  FROM player_activity  
)  
SELECT  
  CORR(total_time_sec, ad_revenue) AS ad_revenue_correlation,  
  CORR(total_time_sec, iap_revenue) AS iap_revenue_correlation,  
  CORR(total_time_sec, total_revenue) AS total_revenue_correlation  
FROM income_analysis  
-- Generated by Cortex Analyst;
```

Result: The total revenue does not depend on time spent playing

A table with three columns: AD_REVENUE_CORRELATION, IAP_REVENUE_CORRELATION, and TOTAL_REVENUE_CORRELATION. The values are 0.1501, 0.0066, and 0.0247 respectively. The table is displayed within a light blue rounded rectangle with a shadow.

AD_REVENUE_CORRELATION	IAP_REVENUE_CORRELATION	TOTAL_REVENUE_CORRELATION
0.1501	0.0066	0.0247

3. Prompt

On which dates in the entire available time period did the daily revenue differ from the average monthly revenue by 10%? By how much %?

Answer:

```
WITH __player_activity_table AS (
  SELECT
    user_id,
    login_date,
    ad_revenue,
    iap_revenue
  FROM cortex_analyst_semantics.semantic_model_generator.player_activity_table
), __player_data_table AS (
  SELECT
    user_id
  FROM cortex_analyst_semantics.semantic_model_generator.player_data_table
), monthly_revenue AS (
  SELECT
    DATE_TRUNC('MONTH', pa.login_date) AS month,
    AVG(pa.ad_revenue + pa.iap_revenue) AS avg_monthly_revenue
  FROM __player_activity_table AS pa
  INNER JOIN __player_data_table AS pd
    ON pa.user_id = pd.user_id
  GROUP BY
    DATE_TRUNC('MONTH', pa.login_date)
), daily_revenue AS (
  SELECT
    pa.login_date AS date,
    SUM(pa.ad_revenue + pa.iap_revenue) AS daily_revenue
  FROM __player_activity_table AS pa
  INNER JOIN __player_data_table AS pd
    ON pa.user_id = pd.user_id
  GROUP BY
    pa.login_date
)
SELECT
  dr.date,
  dr.daily_revenue,
  mr.avg_monthly_revenue,
  (
    dr.daily_revenue - mr.avg_monthly_revenue
  ) / NULLIF(mr.avg_monthly_revenue, 0) * 100 AS percentage_difference
FROM daily_revenue AS dr
INNER JOIN monthly_revenue AS mr
  ON DATE_TRUNC('MONTH', dr.date) = mr.month
WHERE
  ABS(
```

```
(
  dr.daily_revenue - mr.avg_monthly_revenue
) / NULLIF(mr.avg_monthly_revenue, 0)
) > 0.1
ORDER BY
  dr.date DESC
-- Generated by Cortex Analyst;
```

Result: The daily revenue was higher by 10% from the avg month revenue on 19 and 21 February

DATE	DAILY_REVENUE	AVG_MONTHLY_REVENUE	↓ PERCENTAGE_DIFFERENCE
2021-02-21	2,450	0.7471	327,843.8988
2021-02-19	2,215	0.7471	296,388.0555
2021-03-31	1,516	0.6356	238,431.677
2021-01-06	1,257	0.7434	168,979.7863
2021-04-18	1,435	0.9033	158,770.041
2021-03-20	926	0.6356	145,599.4281
2021-08-14	1,201	0.8326	144,144.5119
2021-01-11	1,032	0.7434	138,714.9081
2021-03-24	881	0.6356	138,519.0023

4. Prompt

Compare the total number of installs with the total revenue for each month and calculate revenue per install

Answer:

```
WITH __player_data_table AS (
  SELECT
    user_id,
    install_date
  FROM cortex_analyst_semantics.semantic_model_generator.player_data_table
), __player_activity_table AS (
  SELECT
    user_id,
    login_date,
    ad_revenue,
    iap_revenue
  FROM cortex_analyst_semantics.semantic_model_generator.player_activity_table
), monthly_installs AS (
  SELECT
    DATE_TRUNC('MONTH', pd.install_date) AS month,
    COUNT(DISTINCT pd.user_id) AS total_installs
  FROM __player_data_table AS pd
  GROUP BY
    DATE_TRUNC('MONTH', pd.install_date)
), monthly_revenue AS (
```

```

SELECT
  DATE_TRUNC('MONTH', pa.login_date) AS month,
  SUM(pa.ad_revenue + pa.iap_revenue) AS total_revenue
FROM __player_activity_table AS pa
INNER JOIN __player_data_table AS pd
  ON pa.user_id = pd.user_id
GROUP BY
  DATE_TRUNC('MONTH', pa.login_date)
)
SELECT
  mi.month,
  mi.total_installs,
  mr.total_revenue,
  mr.total_revenue / NULLIF(mi.total_installs, 0) AS revenue_per_install
FROM monthly_installs AS mi
LEFT JOIN monthly_revenue AS mr
  ON mi.month = mr.month
ORDER BY
  mi.month DESC
-- Generated by Cortex Analyst;

```

Result: In May, the highest number of installs and the highest income occurred. In opposite, the lowest income and installs occurred in January. However, the revenue per install does not prove that higher installs generates higher income

MONTH	TOTAL_INSTALLS	TOTAL_REVENUE	REVENUE_PER_INSTALL
2021-12-01	2,150	11,786	5.4819
2021-11-01	1,955	10,523	5.3826
2021-10-01	1,964	10,497	5.3447
2021-09-01	2,963	17,362	5.8596
2021-08-01	2,316	10,993	4.7465
2021-07-01	3,029	14,193	4.6857
2021-06-01	2,891	15,255	5.2767
2021-05-01	3,091	18,393	5.9505
2021-04-01	2,917	13,351	4.577
2021-03-01	3,822	11,140	2.9147
2021-02-01	4,868	12,722	2.6134
2021-01-01	2,904	8,948	3.0813

5. Prompt

What's the biggest difference in days between the install date and the login date? On which device was made? By which user?

Answer:

```
WITH __player_activity_table AS (  
  SELECT  
    user_id,  
    login_date  
  FROM cortex_analyst_semantics.semantic_model_generator.player_activity_table  
) , __player_data_table AS (  
  SELECT  
    user_id,  
    platform,  
    install_date  
  FROM cortex_analyst_semantics.semantic_model_generator.player_data_table  
) , player_activity AS (  
  SELECT  
    pa.user_id,  
    pa.login_date,  
    pd.install_date,  
    pd.platform  
  FROM __player_activity_table AS pa  
  INNER JOIN __player_data_table AS pd  
    ON pa.user_id = pd.user_id  
)  
SELECT  
  user_id,  
  platform,  
  DATEDIFF(DAY, install_date, login_date) AS days_difference  
FROM player_activity  
ORDER BY  
  days_difference DESC NULLS LAST  
LIMIT 1  
-- Generated by Cortex Analyst;
```

Result: The IOS player is the one who waited 360 days to play the game from installing it

What's the biggest difference in days between the install date and the login date?
On which device was it made? By which user?

```
WITH __player_activity_table AS (  
  SELECT  
    user_id,  
    login_date  
  FROM cortex_analyst_semantics.semantic_model_generator.player_  
) , __player_data_table AS (  
  SELECT  
    user_id,  
    platform,  
    install_date  
  FROM cortex_analyst_semantics.semantic_model_generator.player_
```

USER_ID	PLATFORM	DAYS_DIFFERENCE
64bfa653cfa1929c012b2b99c0889192	IOS	360

6. Prompt

Per platform, how many transactions players made and what's the average income from those transactions?

Answer:

```
WITH __player_activity_table AS (  
  SELECT  
    user_id,  
    iap_revenue,  
    transactions  
  FROM cortex_analyst_semantics.semantic_model_generator.player_activity_table  
) , __player_data_table AS (  
  SELECT  
    user_id,  
    platform  
  FROM cortex_analyst_semantics.semantic_model_generator.player_data_table  
) , player_activity AS (  
  SELECT  
    pa.user_id,  
    pd.platform,  
    SUM(pa.transactions) AS total_transactions,  
    AVG(pa.iap_revenue) AS average_iap_revenue  
  FROM __player_activity_table AS pa  
  INNER JOIN __player_data_table AS pd  
    ON pa.user_id = pd.user_id  
  GROUP BY  
    pa.user_id,  
    pd.platform  
)  
SELECT  
  platform,  
  SUM(total_transactions) AS total_transactions,  
  AVG(average_iap_revenue) AS average_iap_revenue  
FROM player_activity  
GROUP BY  
  platform  
-- Generated by Cortex Analyst;
```

Result: Both Android and IOS players made the same number of transactions, but IOS players generated higher avg iap revenue per transaction, even though they are only 6% of the players

PLATFORM	TOTAL_TRANSACTIONS	AVERAGE_IAP_REVENUE
ANDROID	28	0.1924
IOS	28	1.2489

7. Prompt

Do free users play more time (calculated in days), more missions and more sessions than all other users?
Please provide details

Answer:

```
WITH __player_activity_table AS (  
  SELECT  
    user_id,  
    sessions,  
    total_time_sec,  
    missions  
  FROM cortex_analyst_semantics.semantic_model_generator.player_activity_table  
) , __player_data_table AS (  
  SELECT  
    user_id,  
    ua_channel  
  FROM cortex_analyst_semantics.semantic_model_generator.player_data_table  
) , player_activity AS (  
  SELECT  
    pa.user_id,  
    pa.total_time_sec / NULLIF(86400, 0) AS total_time_days,  
    pa.missions,  
    pa.sessions,  
    pd.ua_channel  
  FROM __player_activity_table AS pa  
  INNER JOIN __player_data_table AS pd  
    ON pa.user_id = pd.user_id  
) , organic_users AS (  
  SELECT  
    AVG(total_time_days) AS avg_time_days,  
    AVG(missions) AS avg_missions,  
    AVG(sessions) AS avg_sessions  
  FROM player_activity  
  WHERE  
    ua_channel = 'ORGANIC'  
) , other_users AS (  
  SELECT  
    AVG(total_time_days) AS avg_time_days,  
    AVG(missions) AS avg_missions,  
    AVG(sessions) AS avg_sessions  
  FROM player_activity  
  WHERE  
    ua_channel <> 'ORGANIC'  
)  
SELECT  
  'Organic Users' AS user_type,  
  o.avg_time_days,  
  o.avg_missions,  
  o.avg_sessions
```

```
FROM organic_users AS o
UNION ALL
SELECT
  'Other Users' AS user_type,
  p.avg_time_days,
  p.avg_missions,
  p.avg_sessions
FROM other_users AS p
-- Generated by Cortex Analyst;
```


Result: There is no significant difference in playing more days, sessions or missions between the paid and the free users

Download Search Fullscreen

USER_TYPE	AVG_TIME_DAYS	AVG_MISSIONS	AVG_SESSIONS
Organic Users	107.5917	8.5764	1.8935
Other Users	106.4276	8.114	1.8168

8. Prompt
Can you provide suggestions for questions for this model?

Answer:

 This semantic data model contains information about player activity and player data, including user information, gameplay metrics, and monetization data. You can use this model to analyze player behavior, track revenue and engagement, and gain insights into user acquisition and retention. You can ask questions like 'What is the average daily revenue per user?', 'How many players have completed a certain number of missions?', or 'What is the most popular user acquisition channel?'

Suggestions ^

Calculate total, min, max, and avg revenue per month over the entire available time period, providing the dates on which min and max sales occurred.

Result: Here is the prompt that initially Cortex failed to answer

9. Prompt
Investigate the relationship between the number of missions completed and revenue generated to see if there is a correlation between player progress and revenue.

Answer:
WITH __player_activity_table AS (

```

SELECT
  user_id,
  missions,
  ad_revenue,
  iap_revenue
FROM cortex_analyst_semantics.semantic_model_generator.player_activity_table
), __player_data_table AS (
  SELECT
    user_id
  FROM cortex_analyst_semantics.semantic_model_generator.player_data_table
), combined_data AS (
  SELECT
    pa.user_id,
    pa.missions,
    pa.ad_revenue,
    pa.iap_revenue,
    (
      pa.ad_revenue + pa.iap_revenue
    ) AS total_revenue
  FROM __player_activity_table AS pa
  INNER JOIN __player_data_table AS pd
    ON pa.user_id = pd.user_id
)
SELECT
  missions,
  AVG(total_revenue) AS avg_total_revenue,
  MIN(total_revenue) AS min_total_revenue,
  MAX(total_revenue) AS max_total_revenue
FROM combined_data
GROUP BY
  missions
ORDER BY
  missions
-- Generated by Cortex Analyst;

```

Result: In average, the higher income is generated after the fifth mission is completed

MISSIONS	AVG_TOTAL_REVENUE	MIN_TOTAL_REVENUE	MAX_TOTAL_REVENUE
0	0.536	0	2,284
1	0.2455	0	725
2	0.4159	0	734
3	0.5731	0	737
4	0.7467	0	749
5	0.7084	0	130
6	1.3067	0	780
7	1.1263	0	775
8	1.4031	0	746
9	1.0868	0	21
10	1.2704	0	719

10. Prompt

Identify any outliers in the data, such as players with unusually high revenue or session duration, to understand if there are any unusual patterns in the data that may require further investigation

Answer:

```
WITH __player_activity_table AS (
  SELECT
    user_id,
    login_date,
    sessions,
    total_time_sec,
    missions,
    ad_revenue,
    iap_revenue,
    transactions
  FROM cortex_analyst_semantics.semantic_model_generator.player_activity_table
), __player_data_table AS (
  SELECT
    user_id,
    ua_channel,
    platform,
    install_date
  FROM cortex_analyst_semantics.semantic_model_generator.player_data_table
), player_activity AS (
  SELECT
    pa.user_id,
    pa.login_date,
    pa.sessions,
    pa.total_time_sec,
    pa.missions,
```

```

    pa.ad_revenue,
    pa.iap_revenue,
    pa.transactions,
    pd.ua_channel,
    pd.platform,
    pd.install_date
FROM __player_activity_table AS pa
INNER JOIN __player_data_table AS pd
    ON pa.user_id = pd.user_id
), outliers AS (
SELECT
    user_id,
    login_date,
    sessions,
    total_time_sec,
    missions,
    ad_revenue,
    iap_revenue,
    transactions,
    ua_channel,
    platform,
    install_date,
CASE
    WHEN total_time_sec > (
        SELECT
            AVG(total_time_sec) + 3 * STDDEV(total_time_sec)
        FROM player_activity
    )
    THEN 'High Session Duration'
    WHEN iap_revenue > (
        SELECT
            AVG(iap_revenue) + 3 * STDDEV(iap_revenue)
        FROM player_activity
    )
    THEN 'High IAP Revenue'
    WHEN ad_revenue > (
        SELECT
            AVG(ad_revenue) + 3 * STDDEV(ad_revenue)
        FROM player_activity
    )
    THEN 'High Ad Revenue'
    ELSE 'Normal'
END AS outlier_type
FROM player_activity
)
SELECT
    user_id,
    login_date,
    sessions,

```

```

total_time_sec,
missions,
ad_revenue,
iap_revenue,
transactions,
ua_channel,
platform,
install_date,
outlier_type
FROM outliers
WHERE
    outlier_type <> 'Normal'
ORDER BY
    login_date DESC
-- Generated by Cortex Analyst;

```

Result: There is a need to go in depth with this investigation, since these players did not make any revenue at all, but played longer completing different number of missions

USER_ID	LOGIN_DATE	↓ SESSIONS	TOTAL_TIME_SEC	MISSIONS	AD_REVENUE	IAP_REVENUE	TRANSACTIONS
2642d9a9055beea9fe3153fc925887a	2021-03-27	74	120,210,917	169	0	0	0
91ecfa9650c92d09032831df53bf55ee	2021-08-13	55	101,774,823	84	14	0	0
3ff6f80743f9de71ce44200d662254f4	2021-01-08	46	139,038,686	141	7	0	0
a957477cb288dd30921fea0c6a4624dd	2021-06-17	32	81,060,677	74	21	0	0
c08fcaae1cbea853a9851c8c02aa3ec5	2021-12-04	27	77,203,797	76	13	0	0
e1754e640dedc1e73d5a3d4d410bbd69	2021-02-25	26	182,228,702	227	0	0	0
b940d0ce003c049a7b202f606c0b4fd4	2021-12-08	26	137,950,189	127	20	0	0
54b7247623b9bb0b115bfb6ef664ed78	2021-02-09	25	169,950,161	148	17	0	0

11. Prompt

Analyze the number of sessions and total session duration per month to understand player engagement trends over time.

Answer:

```

WITH __player_activity_table AS (
    SELECT
        user_id,
        login_date,
        sessions,
        total_time_sec
    FROM cortex_analyst_semantics.semantic_model_generator.player_activity_table
), __player_data_table AS (
    SELECT

```

```

    user_id
FROM cortex_analyst_semantics.semantic_model_generator.player_data_table
), monthly_engagement AS (
SELECT
    DATE_TRUNC('MONTH', pa.login_date) AS month,
    COUNT(pa.sessions) AS total_sessions,
    SUM(pa.total_time_sec) AS total_session_duration
FROM __player_activity_table AS pa
INNER JOIN __player_data_table AS pd
    ON pa.user_id = pd.user_id
GROUP BY
    DATE_TRUNC('MONTH', pa.login_date)
)
SELECT
    month,
    total_sessions,
    total_session_duration
FROM monthly_engagement
ORDER BY
    month DESC
-- Generated by Cortex Analyst;

```

Result: In March the players logged in the most and in January played the longest sessions. November is the month with lowest number of sessions played in the shortest time

MONTH	TOTAL_SESSIONS	TOTAL_SESSION_DURATION
2021-12-01	11,008	84,335,317,053
2021-11-01	9,640	77,293,294,495
2021-10-01	10,757	93,987,456,908
2021-09-01	14,513	112,451,508,567
2021-08-01	13,203	115,417,357,679
2021-07-01	14,778	144,321,462,734
2021-06-01	13,636	119,497,935,649
2021-05-01	13,978	112,772,221,748
2021-04-01	14,781	123,524,974,502
2021-03-01	17,528	160,931,895,239
2021-02-01	17,029	147,761,323,192
2021-01-01	12,036	211,618,494,783