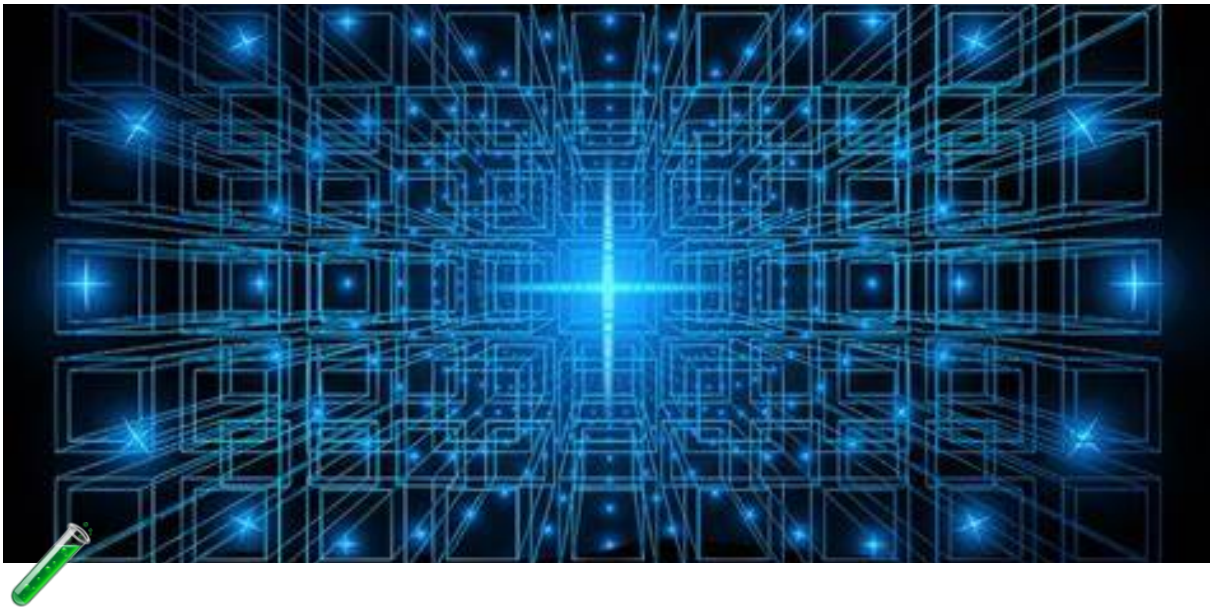


Documento Modelado y Plan de pruebas

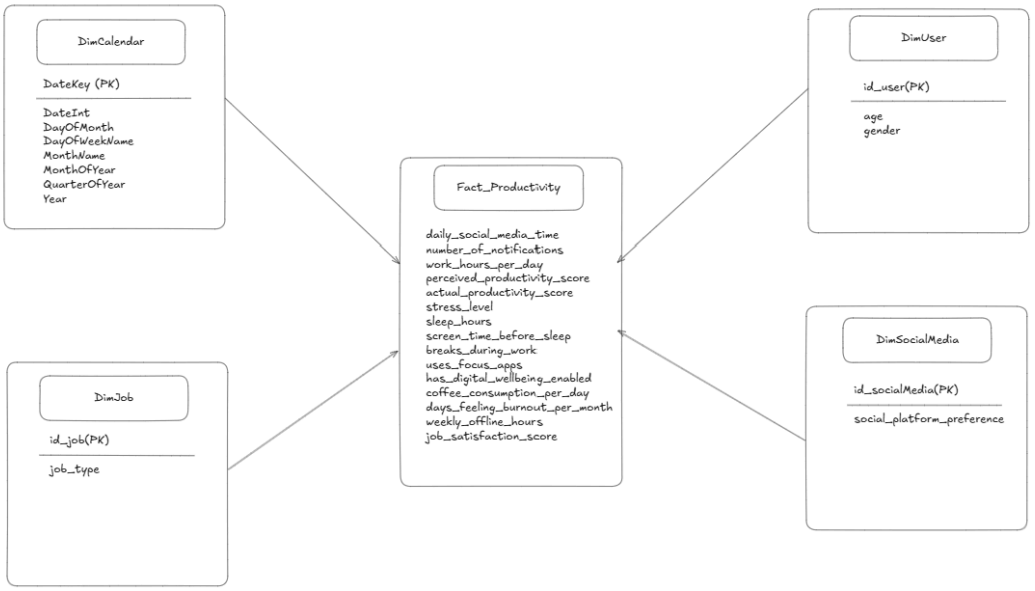
Esquema de estrella

En nuestro caso, queremos predecir la productividad de las personas. Para ello, hemos decidido crear una **tabla de hechos de productividad**, incorporando todos los atributos numéricos que influyen.

Durante el proceso, identificamos varias columnas que podían estructurarse como **tablas de dimensiones**, permitiendo una mejor organización y análisis de los datos. Entre ellas, destacamos:

- **Dimensión de usuario**: Contiene el ID del usuario, su edad y género.
- **Dimensión de red social**: Incluye el ID y el nombre de la red social utilizada.
- **Dimensión de trabajo**: Registra el ID y el nombre de la ocupación del usuario.
- **Dimensión de calendario**: Se utiliza para proporcionar un contexto temporal a los datos de la tabla de hechos, facilitando el análisis de tendencias, comparaciones y agrupaciones en función del tiempo.

Esta estructura nos permitirá realizar análisis más precisos y detectar patrones clave en la productividad de las personas.



MODELO LÓGICO

1. Tabla de hechos: Fact_Productivity

Esta tabla contendrá las métricas y las claves foráneas que conectan con las tablas de dimensiones.

Columna	Tipo de dato	Descripción	Restricción
datekey	INT	Clave foránea a DimCalendar	NOT NULL, FK
id_user	INT	Clave foránea a DimUser	NOT NULL, FK
id_socialMedia	INT	Clave foránea a DimSocialMedia	NOT NULL, FK
id_job	INT	Clave foránea a DimJob	NOT NULL, FK
daily_social_media_time	FLOAT	Tiempo diario en redes sociales2	NOT NULL
number_of_notifications	INT	Número de notificaciones	NOT NULL
work_hours_per_day	FLOAT	Horas de trabajo por día	NOT NULL
perceived_productivity_score	FLOAT	Puntuación de productividad percibida	NOT NULL
actual_productivity_score	FLOAT	Puntuación de productividad real	NOT NULL
stress_level	FLOAT	Nivel de estrés (escala 1-10)	NOT NULL
sleep_hours	FLOAT	Horas de sueño	NOT NULL
screen_time_before_sleep	FLOAT	Tiempo de pantalla antes de dormir	NOT NULL
breaks_during_work	INT	Número de descansos durante el trabajo	NOT NULL
uses_focus_apps	BOOLEAN	Usa aplicaciones de enfoque	NOT NULL
has_digital_wellbeing_enabled	BOOLEAN	Tiene bienestar digital activado	NOT NULL
coffee_consumption_per_day	INT	Consumo de café por día (tazas)	NOT NULL
days_feeling_burnout_per_month	INT	Días sintiendo agotamiento por mes	NOT NULL
weekly_offline_hours	FLOAT	Horas semanales offline	NOT NULL
job_satisfaction_score	FLOAT	Puntuación de satisfacción laboral (1-10)	NOT NULL

2. Tabla de dimensión: DimCalendar

Columna	Tipo de dato	Descripción	Restricción

datekey	INT	Clave primaria	PK, NOT NULL
dateInt	DATE	Fecha en formato YYYY-MM-DD	NOT NULL
dayOfMonth	INT	Día del mes (1-31)	NOT NULL
dayOfWeek_name	VARCHAR(10)	Nombre del día de la semana	NOT NULL
monthName	VARCHAR(10)	Nombre del mes	NOT NULL
monthOfYear	INT	Número del mes (1-12)	NOT NULL
quarterOfYear	INT	Trimestre del año (1-4)	NOT NULL
year	INT	Año	NOT NULL

3. Tabla de dimensión: DimUser

Columna	Tipo de dato	Descripción	Restricción
id_user	INT	Clave primaria	PK, NOT NULL
age	INT	Edad del usuario	NOT NULL
gender	VARCHAR(10)	Género del usuario	NOT NULL

4. Tabla de dimensión: DimSocialMedia

Columna	Tipo de dato	Descripción	Restricción
id_socialMedia	INT	Clave primaria	PK, NOT NULL
social_platform_preference	VARCHAR(50)	Nombre de la red social	NOT NULL

5. Tabla de dimensión: DimJob

Columna	Tipo de dato	Descripción	Restricción
id_job	INT	Clave primaria	PK, NOT NULL
job_type	VARCHAR(50)	Nombre del trabajo	NOT NULL

Plan de Pruebas

Garantizar la calidad, integridad y fiabilidad de los datos del proyecto "Redes Sociales vs Productividad" mediante validaciones sistemáticas aplicadas con PySpark.

Alcance

- **Dataset origen**: Kaggle "Social Media vs Productivity" (30.000 filas, 19 columnas)
- **Tecnología**: PySpark para ingesta y validación
- **Modelo**: Esquema en estrella (1 Fact + 4 Dimensiones)
- **Fecha ejecución**: 11 de junio de 2025

2. DIMENSIONES DE CALIDAD A VALIDAR

UNICIDAD

En este dataset cada fila representa a un único individuo, por lo que no hay problemas de duplicidad en origen. Lo que se hará es generar una clave sintética a partir de cada instancia como clave primaria de la dimensión de usuarios.

EXACTITUD

Comprobamos datos atípicos que sean muy improbables en la realidad, por ejemplo:

- Valores atípicos de horas de sueño (0,1, 100, 24...)
- Valores atípicos de horas de trabajo (34, 70...)

COMPLETITUD

- Comprobar la existencia de datos en blanco, especialmente en campos como perceived_productivity_score
- Identificar si la ausencia de datos es aleatoria o sistemática (Saber la razón de la existencia de los datos vacíos)

PRECISIÓN

- Se convertirán los tipos numéricos a otros más pequeños.

Los valores numéricos del dataset contienen demasiados decimales y rangos de valores demasiado altos, algo que no es necesario y tampoco refleja realmente cómo se hace una medición de este tipo de parámetros.

Ejemplos:

- int64 para edades. Rango numérico innecesario para una edad.
- float64 para medición de productividad. La productividad no se puede medir en el mundo real con tanta precisión decimal, es válido con, como máximo, dos decimales.

CONSISTENCIA

- Verificar formatos consistentes en todas las columnas (tipo de dato).
- Asegurar lógica entre columnas relacionadas (por ejemplo, daily_social_media_time + work_hours_per_day + sleep_hours ≤ 24).

INTEGRIDAD

- Asegurar que todas las claves foráneas y relaciones 1:N en Fact_Productivity tienen sus correspondientes registros y siguen los patrones del modelo dimensional
- Confirmar que todas las PKs en dimensiones son únicas, no nulas y tienen valores apropiados

VALIDEZ

- **Validación de Rangos y Escalas:** Se verifica que los valores numéricos y temporales del dataset se encuentren dentro de límites lógicos y coherentes con la realidad. Esto incluye asegurar que las escalas estén bien definidas (por ejemplo, puntuaciones entre 1 y 10, valores positivos, booleanos correctamente codificados), que las fechas pertenezcan a un rango temporal válido y que variables como horas de actividad diaria, niveles de estrés o satisfacción no excedan los márgenes establecidos.
- **Categorías Permitidas:** Validas que las variables categóricas (`Gender` , `job_type` , etc.) solo contengan valores correctos y normalizados (sin errores de escritura o categorías inventadas).

RAZONABILIDAD

- Correlaciones Lógicas: Analizas si se cumplen relaciones esperadas, como que mucho tiempo en redes sociales implica menor productividad, o que más estrés implica menor satisfacción.

3. CATÁLOGO DE PRUEBAS

3.1 PRUEBAS DE UNICIDAD

ID	Nombre	Tabla	Descripción	Criterio Éxito

El dataset original no incluye una columna de ID de usuario. Cada fila representa una instancia individual recopilada en una encuesta, y **no se garantiza la existencia de un identificador único**.

Por tanto, se ha generado una **clave sintética** combinando atributos como `age`, `gender` y `job_type` para simular una identificación mínima.

Esta clave se ha utilizado únicamente para validar que **no hay duplicados exactos en esos campos**, aunque **dos personas distintas podrían compartir estos mismos valores por coincidencia**.

3.2 PRUEBAS DE EXACTITUD

ID	Nombre	Tabla	Descripción	Criterio Éxito	¿Aceptable?
PE-001	Horas de sueño	Fact_Productivity	Validar que los valores estén dentro del rango lógico (0 < x ≤ 16)	0 fuera de rango	SÍ
PE-002	Coherencia Temporal	Fact_Productivity	Asegurar que work_hours_per_day y screen_time_before_sleep <24 y >0	0 valores imposibles	SÍ

```
# PE-001: Validar horas de sueño fuera de rango
invalid_sleep = dataset.filter((col("sleep_hours") <= 0) | (col("sleep_hours") > 16))
print("PE-001 - Horas de sueño fuera de rango:", invalid_sleep.count())
```

```
PE-001 - Horas de sueño fuera de rango: 0
```

```
# PE-002: Validar coherencia temporal
temporal_issues = dataset.filter(
    (col("work_hours_per_day") > 24) | (col("work_hours_per_day") < 0) |
    (col("screen_time_before_sleep") > 24) | (col("screen_time_before_sleep") < 0)
)
print("PE-002 - Coherencia temporal incorrecta:", temporal_issues.count())
```

```
PE-002 - Coherencia temporal incorrecta: 0
```

3.3 PRUEBAS DE COMPLETITUD

ID	Nombre	Tabla	Descripción	Criterio Éxito	¿Aceptable?
PC-001	Nulos imputados	Fact_Productivity	Verificar los valores nulos o vacíos en columnas	0 nulos	SÍ
PC-002	Saber existencia de datos vacios	Fact_Productivity	Verificar si valores nulos siguen patrones sistemáticos por segmentos (tipo trabajo, edad, plataforma social)	Máximo 15% diferencia entre segmentos	

Nulos imputados:

```python
# Columnas para imputar con mediana (skewness > 0.3)
median_columns = [
    "daily_social_media_time", "screen_time_before_sleep"
]

# Columnas para imputar con media (skewness ≤ 0.3)
mean_columns = [
    "perceived_productivity_score","actual_productivity_score", "stress_level",
    "sleep_hours","job_satisfaction_score"
]

# Imputar con mediana
for col_name in median_columns:
    # El approxQuantile calcula cuantiles aproximados de una columna de forma distribuida
    # El 0.5 indica que queremos el percentil 50, que corresponde a la mediana
    # El 0.05 es el parámetro de error relativo (relative error), que controla la precisión de la aproximación.
    median_value = dfProductivity.approxQuantile(col_name, [0.5], 0.05)[0]
    if median_value is not None:
        dfProductivity = dfProductivity.na.fill({col_name: median_value})
        print(f"Imputado {col_name} con mediana: {median_value:.2f}")

# Imputar con media
for col_name in mean_columns:
    # Aplica la función de agregación avg, que calcula la media, sobre la columna.
    # Hace una coleccion de esos valores y mira el valor columna a columna
    mean_value = dfProductivity.select(col_name).agg({col_name: "avg"}).collect()[0][f"avg({col_name})"]
    if mean_value is not None:
        dfProductivity = dfProductivity.na.fill({col_name: mean_value})
        print(f"Imputado {col_name} con media: {mean_value:.2f}")
```

```
Imputado daily_social_media_time con mediana: 3.00
Imputado screen_time_before_sleep con mediana: 1.00
Imputado perceived_productivity_score con media: 5.51
Imputado actual_productivity_score con media: 4.95
Imputado stress_level con media: 5.51
Imputado sleep_hours con media: 6.50
Imputado job_satisfaction_score con media: 4.96
```

3.4 PRUEBAS DE PRECISIÓN

ID	Nombre	Tabla	Descripción	Criterio Éxito	¿Aceptable?
PP-001	Precisión Decimal	Fact_Productivity	Máximo 2 decimales en scores	Formato correcto	SÍ

```
[ ]  1 dataset.show(5)
```

```
+---+------+----------+----------------------+--------------------------+----------------------+------------------
|age|gender|  job_type|daily_social_media_time|social_platform_preference|number_of_notifications|work_hours_per_day
+---+------+----------+----------------------+--------------------------+----------------------+------------------
| 56|  Male|Unemployed|      4.180939775840407|                  Facebook|                    61|  6.75355840556201
| 46|  Male|    Health|     3.2496029537166797|                   Twitter|                    59|  9.16929611963351
| 32|  Male|   Finance|                  NULL|                   Twitter|                    57| 7.910952034409042
| 60|Female|Unemployed|                  NULL|                  Facebook|                    59| 6.355027219236192
| 25|  Male|        IT|                  NULL|                  Telegram|                    66| 6.214096290148244
+---+------+----------+----------------------+--------------------------+----------------------+------------------
only showing top 5 rows
```

```
1 dataset.show(5)
```

```
+---+------+----------+----------------------+--------------------------+----------------------+---------------+
|age|gender|  job_type|daily_social_media_time|social_platform_preference|number_of_notifications|work_hours_per_day|
+---+------+----------+----------------------+--------------------------+----------------------+---------------+
| 56|  Male|Unemployed|                  4.18|                  Facebook|                    61|           6.75|
| 46|  Male|    Health|                  3.25|                   Twitter|                    59|           9.17|
| 32|  Male|   Finance|                  NULL|                   Twitter|                    57|           7.91|
| 60|Female|Unemployed|                  NULL|                  Facebook|                    59|           6.36|
| 25|  Male|        IT|                  NULL|                  Telegram|                    66|           6.21|
+---+------+----------+----------------------+--------------------------+----------------------+---------------+
only showing top 5 rows
```

3.5 PRUEBAS DE CONSISTENCIA

ID	Nombre	Tabla	Descripción	Criterio Éxito	¿Aceptable?
PCO-001	Logica de caracteristicas	Fact_Productivity	Comprobar la lógica work_hours_per_day +	0 inconsistencias	SÍ

			sleep_hours ≤ 24		

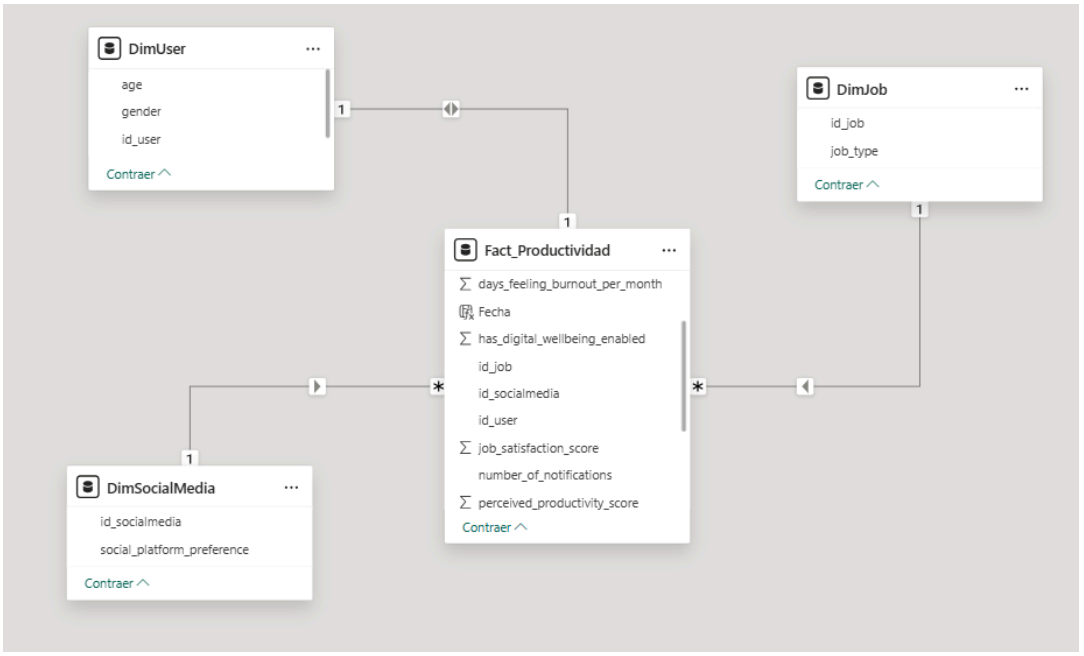
Logica de caracteristicas:

```
# Comprobar la lógica work_hours_per_day + sleep_hours ≤ 24
df = dataset.withColumn(
    "total_hours",
    col("work_hours_per_day") + col("sleep_hours")
)

# Filtrar las filas donde total_hours > 24
df_filtered = df.filter(col("total_hours") > 24)

df_filtered.count()
```

```
0
```

3.6 PRUEBAS DE INTEGRIDAD

ID	Nombre	Tabla	Descripción	Criterio Éxito	¿Aceptable?
PI-001	Integridad Referencial FK	Fact_Productivity	Todas las FK existen en dimensiones padre	0 huérfanos	SÍ, se puede observar que en la dimensión padre existen fk a las dimensiones hijas
PI-002	Coherencia Interna	Fact_Productivity	Apps enfoque vs tiempo RRSS vs productividad	Evaluar patrones	SÍ, los datos muestran patrones coherentes entre apps de enfoque, redes sociales y productividad
PI-003	Cardinalidades Correctas	Fact_Productivity + Dimensiones	Verificar relaciones 1:N esperadas	Cobertura >90% usuarios	SÍ
PI-004	Claves Primarias Válidas	Todas las dimensiones	PKs no nulas y > 0 en todas las dimensiones	Todas PKs válidas	SÍ, Todas son >0 y no hay PKS nulas



3.7 PRUEBAS DE VALIDEZ

ID	Nombre	Tabla	Descripción	Criterio Éxito	¿Aceptable?
PV-001	Validez Numérica	Fact_Productivity	Verificar scores 0-10 (productividad, satisfacción) y stress 1-10 | 0 valores fuera de rango |	Todos los valores son válidos	SÍ
PV-002	Validez Categorica	DimJob, DimUser, DimSocialMedia	Verificar que los valores de las categorias sean los correctos.	Todos los valores son válidos	SÍ
PV-003	Coherencia Temporal	Fact_Productivity	Tiempo en redes sociales ≤ 1440 min (24h), trabajo ≤ 24h, sueño ≤ 24h	0 valores imposibles	SÍ

Validez numérica:

```python
# Validez Numérica (scores 0-10 y stress 1-10)
print("\nValidez: Validando rangos de métricas...")

score_columns = ['perceived_productivity_score', 'actual_productivity_score', 'job_satisfaction_score']
for col_name in score_columns:
    fuera_rango = dataset.filter((col(col_name) < 0) | (col(col_name) > 10)).count()
    if fuera_rango > 0:
        errores.append(f"Validez FALLO - {col_name}: {fuera_rango} valores fuera del rango 0-10")
        print(f"   ❌ {col_name}: {fuera_rango} valores fuera del rango 0-10")
    else:
        print(f"   ✅ {col_name}: todos los valores en rango 0-10")

# Estrés: 1-10
fuera_rango = dataset.filter((col('stress_level') < 1) | (col('stress_level') > 10)).count()
if fuera_rango > 0:
    errores.append(f"Validez FALLO - stress_level: {fuera_rango} valores fuera del rango 1-10")
    print(f"   ❌ stress_level: {fuera_rango} valores fuera del rango 1-10")
else:
    print(f"   ✅ stress_level: todos los valores en rango 1-10")
```

```
Validando rangos de métricas...
   ✅ perceived_productivity_score: todos los valores en rango 0-10
   ✅ actual_productivity_score: todos los valores en rango 0-10
   ✅ job_satisfaction_score: todos los valores en rango 0-10
   ✅ stress_level: todos los valores en rango 1-10
```

Validez categórica:

```python
# Lista de valores esperados
valid_gender = {'Female', 'Other', 'Male'}
valid_job_type = {'Education', 'Student', 'Finance', 'Health', 'IT', 'Unemployed'}
valid_social_platform_preference = {'TikTok', 'Instagram', 'Twitter', 'Telegram', 'Facebook'}

# Validar que todos los valores estén dentro de los valores esperados
dfProductivity_invalid = dfProductivity.filter(
    (~col("gender").isin(*valid_gender)) |
    (~col("job_type").isin(*valid_job_type)) |
    (~col("social_platform_preference").isin(*valid_social_platform_preference))
)

# Mostrar registros inválidos (si hay)
if dfProductivity_invalid.count() > 0:
    dfProductivity_invalid.show()
    print("Se encontraron valores no válidos en las columnas.")
else:
    print("Todos los valores son válidos.")
```

```
Todos los valores son válidos.
```

```
Unique values in column 'gender':
['Female', 'Other', 'Male']
--------------------
Unique values in column 'job_type':
['Education', 'Student', 'Finance', 'Health', 'IT', 'Unemployed']
--------------------
Unique values in column 'social_platform_preference':
['TikTok', 'Instagram', 'Twitter', 'Telegram', 'Facebook']
--------------------
```

```
Unique values in column 'has_digital_wellbeing_enabled':
[1, 0]
--------------------
Unique values in column 'uses_focus_apps':
[1, 0]
--------------------
```

Coherencia Temporal :

```python
# Coherencia Temporal (tiempo pantalla ≤ 24h, trabajo ≤ 24h)
print("\nValidando coherencia temporal...")

# Social media time ≤ 24h
fuera_rango = dataset.filter(col('daily_social_media_time') > 1440).count()
if fuera_rango > 0:
    errores.append(f" FALLO - daily_social_media_time: {fuera_rango} valores > 24h")
    print(f"❌ daily_social_media_time: {fuera_rango} valores > 24h")
else:
    print(f"✅ daily_social_media_time: todos los valores ≤ 24h")

# Horas de trabajo ≤ 24h
fuera_rango = dataset.filter(col('work_hours_per_day') > 24).count()
if fuera_rango > 0:
    errores.append(f"FALLO - work_hours_per_day: {fuera_rango} valores > 24h")
    print(f"❌ work_hours_per_day: {fuera_rango} valores > 24h")
else:
    print(f"✅ work_hours_per_day: todos los valores ≤ 24h")

# Horas de sueño ≤ 24h
fuera_rango = dataset.filter(col('sleep_hours') > 24).count()
if fuera_rango > 0:
    errores.append(f"FALLO - sleep_hours: {fuera_rango} valores > 24h")
    print(f"❌ sleep_hours: {fuera_rango} valores > 24h")
else:
    print(f"✅ sleep_hours: todos los valores ≤ 24h")
```

```
Validando coherencia temporal...
✅ daily_social_media_time: todos los valores ≤ 24h
✅ work_hours_per_day: todos los valores ≤ 24h
✅ sleep_hours: todos los valores ≤ 24h
```

3.8 PRUEBAS DE RAZONABILIDAD

ID	Nombre	Tabla	Descripción	Criterio Éxito	¿Aceptable?
PR-001	Rangos Contextuales	Fact_Productivity	Horas trabajo ≤16h, RRSS ≤12h, notificaciones ≤1000, café ≤15	Máximo 5% irrazonables	SÍ (0%)
PR-003	Distribuciones Esperadas	DimUser, Fact_Productivity	Edades laborales (16-80), ≤25 días/mes, outliers extremos	Máximo 2% outliers	

Distribuciones Esperadas:

```python
# Rangos Contextuales (edad 16-80) con tolerancia a outliers
print("\nValidando rangos contextuales (edad 16-80) con tolerancia del 2% a outliers...")

total_filas = dataset.count()

fuera_rango_count = dataset.filter((col('age') < 16) | (col('age') > 80)).count()

# Calculo del porcenaje de outliers
porcentaje_fuera_rango = (fuera_rango_count / total_filas) * 100 if total_filas > 0 else 0

# Porcentaje de outliers aceptado
outlier_threshold_percent = 2.0

if porcentaje_fuera_rango > outlier_threshold_percent:
    errores.append(f" FALLO - age: {fuera_rango_count} valores ({porcentaje_fuera_rango:.2f}%) fuera del rango 16-80 (excede el {outlier_threshold_percent}%)")
    print(f"    ❌ FALLO - age: {fuera_rango_count} valores ({porcentaje_fuera_rango:.2f}%) fuera del rango 16-80 (excede el {outlier_threshold_percent}%)")
else:
    print(f"    ✅ age: {fuera_rango_count} valores ({porcentaje_fuera_rango:.2f}%) fuera del rango 16-80 (dentro del umbral del {outlier_threshold_percent}%)")
```

```
Validando rangos contextuales (edad 16-80) con tolerancia del 2% a outliers...
    ✅ age: 0 valores (0.00%) fuera del rango 16-80 (dentro del umbral del 2.0%)
```