

DOCUMENTACIÓN DE BASE DE DATOS

Sistema de Test de Personalidad OEJTS (Open Extended Jungian Type Scales)

Versión: 1.0

Fecha: Octubre 2025

Motor de BD: PostgreSQL

Autor: [Tu Nombre]

TABLA DE CONTENIDOS

1. Descripción General del Sistema
2. Modelo Entidad-Relación
3. Diccionario de Datos
4. Algoritmo de Cálculo
5. Reglas de Negocio
6. Índices y Optimización
7. Seguridad y Consideraciones

1. DESCRIPCIÓN GENERAL DEL SISTEMA

1.1 Objetivo

El sistema tiene como objetivo proporcionar una plataforma web para administrar y realizar el test de personalidad OEJTS (Open Extended Jungian Type Scales), recolectando respuestas de usuarios y generando perfiles de personalidad basados en la teoría de tipos psicológicos de Carl Jung.

1.2 Flujo del Sistema

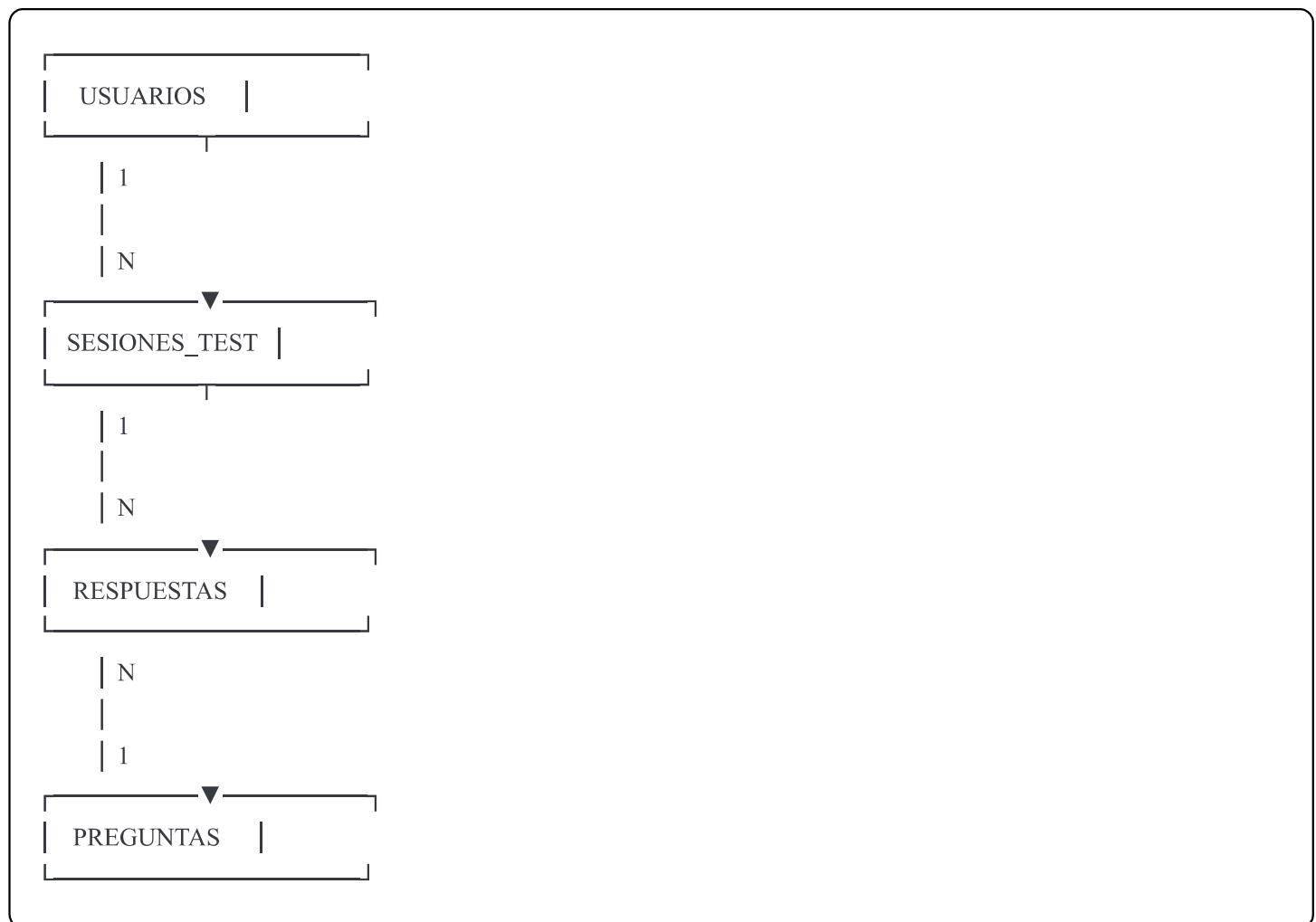
1. **Usuario anónimo** accede al sistema y comienza el test
2. Usuario responde las **32 preguntas** del test OEJTS
3. Al completar el test y solicitar resultados, el sistema **requiere registro**
4. Usuario se registra con sus credenciales
5. Usuario inicia sesión
6. Sistema calcula y muestra **resultados detallados** del test
7. Usuario puede visualizar su perfil de personalidad completo

1.3 Características del Test OEJTS

- **32 preguntas** con escala Likert de 5 puntos (1-5)
- Mide **4 dimensiones** de personalidad:
 - **I/E:** Introversión vs Extraversión
 - **S/N:** Sensación vs Intuición
 - **F/T:** Sentimiento vs Pensamiento
 - **J/P:** Juicio vs Percepción
- Genera **16 tipos de personalidad** posibles (combinación de 4 letras)

2. MODELO ENTIDAD-RELACIÓN

2.1 Diagrama Conceptual



2.2 Relaciones

- Un **USUARIO** puede tener múltiples **SESIONES_TEST** (1:N)
- Una **SESION_TEST** pertenece a un único **USUARIO** (N:1)
- Una **SESION_TEST** tiene múltiples **RESPUESTAS** (1:N)

- Una **RESPUESTA** pertenece a una única **SESION_TEST** (N:1)
 - Una **RESPUESTA** referencia a una única **PREGUNTA** (N:1)
 - Una **PREGUNTA** puede tener múltiples **RESPUESTAS** de diferentes sesiones (1:N)
-

3. DICCCIONARIO DE DATOS

3.1 TABLA: usuarios

Descripción: Almacena la información de los usuarios registrados en el sistema.

Campo	Tipo	Restricciones	Descripción
id_usuario	SERIAL	PRIMARY KEY	Identificador único del usuario
nombre_completo	VARCHAR(150)	NOT NULL	Nombre completo del usuario
email	VARCHAR(100)	NOT NULL, UNIQUE	Correo electrónico (credencial de acceso)
password_hash	VARCHAR(255)	NOT NULL	Contraseña encriptada (bcrypt/argon2)
fecha_registro	TIMESTAMP	NOT NULL, DEFAULT NOW()	Fecha y hora de registro
ultimo_acceso	TIMESTAMP	NULL	Última fecha de inicio de sesión
activo	BOOLEAN	NOT NULL, DEFAULT TRUE	Estado de la cuenta
fecha_nacimiento	DATE	NULL	Fecha de nacimiento del usuario
genero	VARCHAR(20)	NULL	Género del usuario (opcional)
pais	VARCHAR(50)	NULL	País de residencia (opcional)

Índices:

- PRIMARY KEY en **[id_usuario]**
 - UNIQUE INDEX en **[email]**
 - INDEX en **[fecha_registro]** para consultas temporales
-

3.2 TABLA: preguntas

Descripción: Contiene las 32 preguntas del test OEJTS con sus pesos para el cálculo.

Campo	Tipo	Restricciones	Descripción
id_pregunta	INTEGER	PRIMARY KEY	Número de pregunta (1-32)
texto_izquierda	VARCHAR(200)	NOT NULL	Descripción del extremo izquierdo
texto_derecha	VARCHAR(200)	NOT NULL	Descripción del extremo derecho
dimension	VARCHAR(2)	NOT NULL	Dimensión que mide: IE, SN, FT, JP
peso	INTEGER	NOT NULL	Peso en el cálculo (+1 o -1)
orden	INTEGER	NOT NULL, UNIQUE	Orden de presentación
activa	BOOLEAN	NOT NULL, DEFAULT TRUE	Si la pregunta está activa

Valores de la dimensión:

- **[IE]**: Introversión/Extraversión
- **[SN]**: Sensación/Intuición
- **[FT]**: Sentimiento/Pensamiento
- **[JP]**: Juicio/Percepción

Índices:

- PRIMARY KEY en **[id_pregunta]**
 - INDEX en **[dimension]**
 - UNIQUE INDEX en **[orden]**
-

3.3 TABLA: sesiones_test

Descripción: Registra cada intento de test realizado por un usuario.

Campo	Tipo	Restricciones	Descripción
id_sesion	SERIAL	PRIMARY KEY	Identificador único de la sesión
id_usuario	INTEGER	FOREIGN KEY, NULL	Usuario que realizó el test (NULL si anónimo)
token_anonimo	VARCHAR(64)	UNIQUE	Token para usuarios no registrados
fecha_inicio	TIMESTAMP	NOT NULL, DEFAULT NOW()	Inicio del test
fecha_fin	TIMESTAMP	NULL	Finalización del test
completado	BOOLEAN	NOT NULL, DEFAULT FALSE	Si el test fue completado
resultado_ie	INTEGER	NULL	Puntuación IE calculada
resultado_sn	INTEGER	NULL	Puntuación SN calculada
resultado_ft	INTEGER	NULL	Puntuación FT calculada
resultado_jp	INTEGER	NULL	Puntuación JP calculada
tipo_personalidad	VARCHAR(4)	NULL	Tipo resultante (ej: INTJ, ESFP)
ip_address	VARCHAR(45)	NULL	IP del usuario (IPv4/IPv6)
user_agent	TEXT	NULL	Navegador y dispositivo

Índices:

- PRIMARY KEY en `id_sesion`
- FOREIGN KEY en `id_usuario` → `usuarios(id_usuario)`
- UNIQUE INDEX en `token_anonimo`
- INDEX en `fecha_inicio`
- INDEX en `tipo_personalidad`

Reglas:

- Un registro puede tener `id_usuario` NULL y `token_anonimo` NOT NULL (sesión anónima)
- Al registrarse, se actualiza `id_usuario` y se mantiene `token_anonimo`

3.4 TABLA: respuestas

Descripción: Almacena cada respuesta individual del test.

Campo	Tipo	Restricciones	Descripción
id_respuesta	SERIAL	PRIMARY KEY	Identificador único de la respuesta
id_sesion	INTEGER	FOREIGN KEY, NOT NULL	Sesión a la que pertenece
id_pregunta	INTEGER	FOREIGN KEY, NOT NULL	Pregunta respondida
valor_respuesta	INTEGER	NOT NULL, CHECK (1-5)	Valor seleccionado (1-5)
tiempo_respuesta	INTEGER	NULL	Tiempo en segundos
fecha_respuesta	TIMESTAMP	NOT NULL, DEFAULT NOW()	Momento de la respuesta

Índices:

- PRIMARY KEY en `id_respuesta`
- FOREIGN KEY en `id_sesion` → `sesiones_test(id_sesion)`
- FOREIGN KEY en `id_pregunta` → `preguntas(id_pregunta)`
- UNIQUE INDEX en `(id_sesion, id_pregunta)` - Una respuesta por pregunta por sesión
- INDEX en `id_sesion` para consultas rápidas

Constraints:

- CHECK: `valor_respuesta BETWEEN 1 AND 5`
-

3.5 TABLA: tipos_personalidad

Descripción: Catálogo con las descripciones de los 16 tipos de personalidad.

Campo	Tipo	Restricciones	Descripción
codigo	VARCHAR(4)	PRIMARY KEY	Código del tipo (INTJ, ESFP, etc.)
nombre	VARCHAR(100)	NOT NULL	Nombre descriptivo del tipo
descripcion_corta	TEXT	NOT NULL	Resumen del tipo
descripcion_completa	TEXT	NOT NULL	Descripción detallada
fortalezas	TEXT	NULL	Fortalezas principales
debilidades	TEXT	NULL	Áreas de mejora
carreras_sugeridas	TEXT	NULL	Profesiones recomendadas
famosos_tipo	TEXT	NULL	Personas famosas con este tipo
porcentaje_poblacion	DECIMAL(4,2)	NULL	% de la población mundial

Índices:

- PRIMARY KEY en `codigo`

Datos: 16 registros fijos (INTJ, INTP, ENTJ, ENTP, INFJ, INFP, ENFJ, ENFP, ISTJ, ISFJ, ESTJ, ESFJ, ISTP, ISFP, ESTP, ESFP)

4. ALGORITMO DE CÁLCULO

4.1 Fórmulas de Cálculo

El test OEJTS calcula 4 puntuaciones basándose en las respuestas del usuario. Cada dimensión tiene una fórmula específica que suma o resta valores según las preguntas correspondientes.

4.1.1 Dimensión IE (Introversión/Extraversión)

$$IE = 30 - Q3 - Q7 - Q11 + Q15 - Q19 + Q23 + Q27 - Q31$$

Interpretación:

- Si $IE > 24 \rightarrow$ **Extravertido (E)**
- Si $IE \leq 24 \rightarrow$ **Introvertido (I)**

Preguntas involucradas:

Pregunta	Peso	Descripción
Q3	-1	aburrido estando solo / necesita tiempo solo
Q7	-1	energético / tranquilo
Q11	-1	trabaja mejor en grupos / trabaja mejor solo
Q15	+1	se agota en fiestas / se energiza en fiestas
Q19	-1	habla más / escucha más
Q23	+1	se queda en casa / sale a la ciudad
Q27	+1	le cuesta gritar fuerte / gritar viene naturalmente
Q31	-1	le gusta actuar frente a otros / evita hablar en público

4.1.2 Dimensión SN (Sensación/Intuición)

$$SN = 12 + Q4 + Q8 + Q12 + Q16 + Q20 - Q24 - Q28 + Q32$$

Interpretación:

- Si $SN > 24 \rightarrow$ **Intuitivo (N)**
- Si $SN \leq 24 \rightarrow$ **Sensorial (S)**

Preguntas involucradas:

Pregunta	Peso	Descripción
Q4	+1	acepta las cosas como son / insatisfecho con cómo son
Q8	+1	prefiere examen opción múltiple / prefiere ensayos
Q12	+1	enfocado en el presente / enfocado en el futuro
Q16	+1	encaja / sobresale
Q20	+1	describe qué pasó / describe qué significó
Q24	-1	quiere la visión general / quiere los detalles
Q28	-1	teórico / empírico
Q32	+1	le gusta saber quién, qué, cuándo / le gusta saber por qué

4.1.3 Dimensión FT (Sentimiento/Pensamiento)

$$FT = 30 - Q2 + Q6 + Q10 - Q14 - Q18 + Q22 - Q26 - Q30$$

Interpretación:

- Si $FT > 24 \rightarrow$ Pensamiento (T)
- Si $FT \leq 24 \rightarrow$ Sentimiento (F)

Preguntas involucradas:

Pregunta	Peso	Descripción
Q2	-1	escéptico / quiere creer
Q6	+1	piensa que "robótico" es insulto / aspira a mente mecánica
Q10	+1	se lastima fácilmente / piel gruesa
Q14	-1	quiere respeto de la gente / quiere su amor
Q18	-1	quiere ser bueno arreglando cosas / arreglando personas
Q22	+1	sigue el corazón / sigue la cabeza
Q26	-1	basa moralidad en justicia / en compasión
Q30	-1	incómodo con emociones / valora las emociones

4.1.4 Dimensión JP (Juicio/Percepción)

$$JP = 18 + Q1 + Q5 - Q9 + Q13 - Q17 + Q21 - Q25 + Q29$$

Interpretación:

- Si $JP > 24 \rightarrow$ Perceptivo (P)
- Si $JP \leq 24 \rightarrow$ Juicioso (J)

Preguntas involucradas:

Pregunta	Peso	Descripción
Q1	+1	hace listas / confía en la memoria
Q5	+1	mantiene cuarto limpio / deja cosas donde sea
Q9	-1	caótico / organizado
Q13	+1	planea con anticipación / planea último minuto
Q17	-1	mantiene opciones abiertas / se compromete
Q21	+1	hace el trabajo de inmediato / procrastina
Q25	-1	improvisa / prepara
Q29	+1	trabaja duro / juega duro

4.2 Proceso de Cálculo Completo

Paso 1: Recolección de Respuestas

```
sql  
-- Obtener todas las respuestas de una sesión  
SELECT id_pregunta, valor_respuesta  
FROM respuestas  
WHERE id_sesion = [ID_SESION]  
ORDER BY id_pregunta;
```

Paso 2: Aplicar Fórmulas

Para cada dimensión, aplicar la fórmula correspondiente usando los valores de respuesta (1-5).

Ejemplo de cálculo IE:

Supongamos las respuestas:

Q3 = 4, Q7 = 2, Q11 = 3, Q15 = 4, Q19 = 2, Q23 = 5, Q27 = 3, Q31 = 1

$$IE = 30 - 4 - 2 - 3 + 4 - 2 + 5 + 3 - 1$$

$$IE = 30 - 12 + 12$$

$$IE = 30$$

Como IE (30) > 24 → El usuario es EXTRAVERTIDO (E)

Paso 3: Determinar Tipo de Personalidad

```
IF IE > 24 THEN letra1 = 'E' ELSE letra1 = 'T'  
IF SN > 24 THEN letra2 = 'N' ELSE letra2 = 'S'  
IF FT > 24 THEN letra3 = 'T' ELSE letra3 = 'F'  
IF JP > 24 THEN letra4 = 'P' ELSE letra4 = 'J'  
  
tipo_personalidad = letra1 + letra2 + letra3 + letra4
```

Paso 4: Almacenar Resultados

```
sql  
  
UPDATE sesiones_test  
SET  
    resultado_ie = [VALOR_IE],  
    resultado_sn = [VALOR_SN],  
    resultado_ft = [VALOR_FT],  
    resultado_jp = [VALOR_JP],  
    tipo_personalidad = [TIPO],  
    fecha_fin = NOW(),  
    completado = TRUE  
WHERE id_sesion = [ID_SESION];
```

4.3 Validaciones

Antes de calcular:

1. Verificar que existan exactamente 32 respuestas
2. Verificar que todas las respuestas estén entre 1 y 5
3. Verificar que no haya respuestas duplicadas para la misma pregunta

Después de calcular:

1. Verificar que todas las puntuaciones estén en rango válido
2. Verificar que el tipo de personalidad sea uno de los 16 válidos
3. Registrar la fecha de finalización

5. REGLAS DE NEGOCIO

5.1 Gestión de Usuarios

RN-001: Registro de Usuario

- El email debe ser único en el sistema
- El password debe tener mínimo 8 caracteres

- El password debe ser encriptado antes de almacenarse
- La fecha de registro se asigna automáticamente

RN-002: Inicio de Sesión

- Validar email y password
- Actualizar campo `ultimo_acceso` al iniciar sesión
- Solo usuarios activos pueden iniciar sesión

RN-003: Asociación de Sesión Anónima

- Al registrarse, buscar sesión con `token_anonimo` de la sesión actual
- Asociar esa sesión al nuevo `id_usuario`
- Mantener `token_anonimo` para trazabilidad

5.2 Gestión de Test

RN-004: Inicio de Test

- Crear registro en `sesiones_test` con `token_anonimo` único
- `completado = FALSE` por defecto
- Registrar `fecha_inicio`, `ip_address`, `user_agent`

RN-005: Respuestas

- Solo se permite una respuesta por pregunta por sesión
- Valor de respuesta debe estar entre 1 y 5
- No se puede modificar una respuesta una vez guardada (opcional)
- Registrar `tiempo_respuesta` para análisis

RN-006: Finalización de Test

- El test está completo cuando hay 32 respuestas
- No se pueden calcular resultados con test incompleto
- Al finalizar, ejecutar algoritmo de cálculo
- Almacenar resultados en `sesiones_test`

RN-007: Visualización de Resultados

- Solo usuarios registrados y autenticados pueden ver resultados
- El usuario solo puede ver sus propias sesiones
- Los resultados incluyen:
 - Tipo de personalidad (4 letras)

- Puntuaciones de cada dimensión
- Descripción completa del tipo
- Gráficos comparativos

5.3 Privacidad y Datos

RN-008: Protección de Datos

- Los passwords NUNCA se almacenan en texto plano
- Las IPs se almacenan pero no se muestran al usuario
- Los datos personalesopcionales (género, país) son privados

RN-009: Anonimato Previo

- Las sesiones anónimas no están vinculadas a usuario hasta el registro
 - El `token_anonimo` permite la trazabilidad sin identificación
-

6. ÍNDICES Y OPTIMIZACIÓN

6.1 Índices Primarios

```
sql

-- Tabla usuarios
CREATE INDEX idx_usuarios_email ON usuarios(email);
CREATE INDEX idx_usuarios_fecha_registro ON usuarios(fecha_registro);

-- Tabla sesiones_test
CREATE INDEX idx_sesiones_usuario ON sesiones_test(id_usuario);
CREATE INDEX idx_sesiones_token ON sesiones_test(token_anonimo);
CREATE INDEX idx_sesiones_tipo ON sesiones_test(tipo_personalidad);
CREATE INDEX idx_sesiones_fecha ON sesiones_test(fecha_inicio);

-- Tabla respuestas
CREATE INDEX idx_respuestas_sesion ON respuestas(id_sesion);
CREATE INDEX idx_respuestas_pregunta ON respuestas(id_pregunta);
CREATE UNIQUE INDEX idx_respuestas_sesion_pregunta
    ON respuestas(id_sesion, id_pregunta);
```

6.2 Consultas Frecuentes Optimizadas

Obtener todas las sesiones de un usuario con resultados:

```
sql
```

```
SELECT s.id_sesion, s.fecha_inicio, s.fecha_fin, s.tipo_personalidad,  
    s.resultado_ie, s.resultado_sn, s.resultado_ft, s.resultado_jp  
FROM sesiones_test s  
WHERE s.id_usuario = [ID_USUARIO]  
AND s.completado = TRUE  
ORDER BY s.fecha_inicio DESC;
```

Obtener respuestas de una sesión:

sql

```
SELECT r.id_pregunta, r.valor_respuesta,  
    p.texto_izquierda, p.texto_derecha  
FROM respuestas r  
INNER JOIN preguntas p ON r.id_pregunta = p.id_pregunta  
WHERE r.id_sesion = [ID_SESION]  
ORDER BY r.id_pregunta;
```

Estadísticas de tipos de personalidad:

sql

```
SELECT tipo_personalidad, COUNT(*) as cantidad,  
    ROUND(COUNT(*) * 100.0 / SUM(COUNT(*)) OVER(), 2) as porcentaje  
FROM sesiones_test  
WHERE completado = TRUE AND tipo_personalidad IS NOT NULL  
GROUP BY tipo_personalidad  
ORDER BY cantidad DESC;
```

7. SEGURIDAD Y CONSIDERACIONES

7.1 Seguridad de Datos

Encriptación de Contraseñas:

- Usar bcrypt o Argon2 con salt
- Nunca almacenar contraseñas en texto plano
- Nunca enviar contraseñas por logs o errores

Protección contra Inyección SQL:

- Usar consultas preparadas (prepared statements)
- Validar y sanitizar todos los inputs

- Usar ORMs cuando sea posible

Control de Acceso:

- Implementar autenticación basada en sesiones/JWT
- Verificar permisos antes de cada operación
- Los usuarios solo acceden a sus propios datos

7.2 GDPR y Privacidad

Datos Personales:

- Email, nombre, fecha de nacimiento son datos personales
- IP address se considera dato personal
- Implementar consentimiento explícito

Derechos del Usuario:

- Derecho al acceso: exportar todos sus datos
- Derecho al olvido: eliminar cuenta y datos
- Derecho a rectificación: modificar datos personales

7.3 Respaldo y Recuperación

Respaldos:

- Backup diario de toda la base de datos
- Backup incremental cada 6 horas
- Retención de backups: 30 días mínimo

Recuperación:

- Procedimiento documentado de restauración
- Pruebas periódicas de recuperación
- Plan de contingencia ante pérdida de datos

8. DATOS DE PRUEBA Y CATÁLOGOS

8.1 Catálogo de Preguntas (32 preguntas)

```
sql
```

```

INSERT INTO preguntas (id_pregunta, texto_izquierda, texto_derecha, dimension, peso, orden) VALUES
(1, 'hace listas', 'confía en la memoria', 'JP', 1, 1),
(2, 'escéptico', 'quiere creer', 'FT', -1, 2),
(3, 'aburrido estando solo', 'necesita tiempo solo', 'IE', -1, 3),
(4, 'acepta las cosas como son', 'insatisfecho con cómo son las cosas', 'SN', 1, 4),
(5, 'mantiene un cuarto limpio', 'deja cosas donde sea', 'JP', 1, 5),
(6, 'piensa que "robótico" es un insulto', 'aspira a tener una mente mecánica', 'FT', 1, 6),
(7, 'energético', 'tranquilo', 'IE', -1, 7),
(8, 'prefiere examen de opción múltiple', 'prefiere respuestas de ensayo', 'SN', 1, 8),
(9, 'caótico', 'organizado', 'JP', -1, 9),
(10, 'se lastima fácilmente', 'piel gruesa', 'FT', 1, 10),
(11, 'trabaja mejor en grupos', 'trabaja mejor solo', 'IE', -1, 11),
(12, 'enfocado en el presente', 'enfocado en el futuro', 'SN', 1, 12),
(13, 'planea con mucha anticipación', 'planea a último minuto', 'JP', 1, 13),
(14, 'quiere el respeto de la gente', 'quiere su amor', 'FT', -1, 14),
(15, 'se agota en fiestas', 'se energiza en fiestas', 'IE', 1, 15),
(16, 'encaja', 'sobresale', 'SN', 1, 16),
(17, 'mantiene opciones abiertas', 'se compromete', 'JP', -1, 17),
(18, 'quiere ser bueno arreglando cosas', 'quiere ser bueno arreglando personas', 'FT', -1, 18),
(19, 'habla más', 'escucha más', 'IE', -1, 19),
(20, 'al describir un evento, dice qué pasó', 'al describir un evento, dice qué significó', 'SN', 1, 20),
(21, 'hace el trabajo de inmediato', 'procrastina', 'JP', 1, 21),
(22, 'sigue el corazón', 'sigue la cabeza', 'FT', 1, 22),
(23, 'se queda en casa', 'sale a la ciudad', 'IE', 1, 23),
(24, 'quiere la visión general', 'quiere los detalles', 'SN', -1, 24),
(25, 'improvisa', 'prepara', 'JP', -1, 25),
(26, 'basa la moralidad en la justicia', 'basa la moralidad en la compasión', 'FT', -1, 26),
(27, 'le cuesta gritar muy fuerte', 'gritar a otros cuando están lejos viene naturalmente', 'IE', 1, 27),
(28, 'teórico', 'empírico', 'SN', -1, 28),
(29, 'trabaja duro', 'juega duro', 'JP', 1, 29),
(30, 'incómodo con las emociones', 'valora las emociones', 'FT', -1, 30),
(31, 'le gusta actuar frente a otras personas', 'evita hablar en público', 'IE', -1, 31),
(32, 'le gusta saber "quién", "qué", "cuándo"', 'le gusta saber "por qué"', 'SN', 1, 32);

```

8.2 Catálogo de Tipos de Personalidad

Los 16 tipos: **INTJ, INTP, ENTJ, ENTP, INFJ, INFP, ENFJ, ENFP, ISTJ, ISFJ, ESTJ, ESFJ, ISTP, ISFP, ESTP, ESFP**

Cada tipo debe tener su descripción completa en la tabla **tipos_personalidad**.

9. APÉNDICES

9.1 Esquema de Valores

Escala de Respuestas:

- 1 = Totalmente identificado con lado izquierdo
- 2 = Parcialmente identificado con lado izquierdo
- 3 = Neutral / Punto medio
- 4 = Parcialmente identificado con lado derecho
- 5 = Totalmente identificado con lado derecho

Rangos de Puntuación:

- IE, FT: 8 a 38 (punto de corte: 24)
- SN: 4 a 44 (punto de corte: 24)
- JP: 10 a 40 (punto de corte: 24)

9.2 Ejemplo de Sesión Completa

Escenario: Usuario anónimo completa test y se registra

1. Usuario inicia test → crea `sesion_test` con `token_anonimo = "abc123"`
2. Usuario responde 32 preguntas → 32 registros en `respuestas`
3. Usuario solicita resultados → sistema requiere registro
4. Usuario se registra → crea `usuario` con `id_usuario = 1`
5. Sistema asocia → actualiza `sesiones_test` SET `id_usuario = 1` WHERE `token_anonimo = "abc123"`
6. Sistema calcula resultados usando el algoritmo
7. Sistema actualiza `sesiones_test` con resultados y `completado = TRUE`
8. Usuario inicia sesión y visualiza resultados completos

Estado Final de los Datos:

```
sql
```

```
-- Tabla usuarios
id_usuario: 1
email: "usuario@ejemplo.com"
nombre_completo: "Juan Pérez"
fecha_registro: "2025-10-05 14:30:00"

-- Tabla sesiones_test
id_sesion: 1
id_usuario: 1
token_anonimo: "abc123"
fecha_inicio: "2025-10-05 14:00:00"
fecha_fin: "2025-10-05 14:25:00"
completado: TRUE
resultado_ie: 28
resultado_sn: 26
resultado_ft: 22
resultado_jp: 30
tipo_personalidad: "ENTP"
```

-- Tabla respuestas (32 registros)

Ejemplo:

```
id_respuesta: 1, id_sesion: 1, id_pregunta: 1, valor_respuesta: 3
id_respuesta: 2, id_sesion: 1, id_pregunta: 2, valor_respuesta: 4
...
id_respuesta: 32, id_sesion: 1, id_pregunta: 32, valor_respuesta: 5
```

10. SCRIPTS SQL DE CREACIÓN

10.1 Script Completo de Creación de Base de Datos

sql

```

-- =====
-- CREACIÓN DE BASE DE DATOS: OEJTS
-- Sistema de Test de Personalidad
-- Motor: PostgreSQL 14+
-- =====

-- Crear base de datos
CREATE DATABASE oejts_db
ENCODING = 'UTF8'
LC_COLLATE = 'es_ES.UTF-8'
LC_CTYPE = 'es_ES.UTF-8';

\c oejts_db;

-- =====
-- TABLA: usuarios
-- =====

CREATE TABLE usuarios (
    id_usuario SERIAL PRIMARY KEY,
    nombre_completo VARCHAR(150) NOT NULL,
    email VARCHAR(100) NOT NULL UNIQUE,
    password_hash VARCHAR(255) NOT NULL,
    fecha_registro TIMESTAMP NOT NULL DEFAULT NOW(),
    ultimo_acceso TIMESTAMP NULL,
    activo BOOLEAN NOT NULL DEFAULT TRUE,
    fecha_nacimiento DATE NULL,
    genero VARCHAR(20) NULL,
    pais VARCHAR(50) NULL,
    CONSTRAINT chk_email_formato CHECK (email ~* '^[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,}')
);

-- Índices de usuarios
CREATE INDEX idx_usuarios_email ON usuarios(email);
CREATE INDEX idx_usuarios_fecha_registro ON usuarios(fecha_registro);
CREATE INDEX idx_usuarios_activo ON usuarios(activo);

-- Comentarios de tabla usuarios
COMMENT ON TABLE usuarios IS 'Almacena información de usuarios registrados';
COMMENT ON COLUMN usuarios.password_hash IS 'Contraseña encriptada con bcrypt o argon2';
COMMENT ON COLUMN usuarios.activo IS 'FALSE para cuentas desactivadas o eliminadas';

-- =====
-- TABLA: preguntas
-- =====

CREATE TABLE preguntas (

```

```

id_pregunta INTEGER PRIMARY KEY,
texto_izquierda VARCHAR(200) NOT NULL,
texto_derecha VARCHAR(200) NOT NULL,
dimension VARCHAR(2) NOT NULL,
peso INTEGER NOT NULL,
orden INTEGER NOT NULL UNIQUE,
activa BOOLEAN NOT NULL DEFAULT TRUE,

CONSTRAINT chk_dimension CHECK (dimension IN ('IE', 'SN', 'FT', 'JP')),
CONSTRAINT chk_peso CHECK (peso IN (-1, 1)),
CONSTRAINT chk_id_pregunta_rango CHECK (id_pregunta BETWEEN 1 AND 32)
);

-- Índices de preguntas
CREATE INDEX idx_preguntas_dimension ON preguntas(dimension);
CREATE INDEX idx_preguntas_orden ON preguntas(orden);
CREATE INDEX idx_preguntas_activa ON preguntas(activa);

-- Comentarios de tabla preguntas
COMMENT ON TABLE preguntas IS 'Catálogo de las 32 preguntas del test OEJTS';
COMMENT ON COLUMN preguntas.dimension IS 'IE=Intro/Extra, SN=Sens/Intui, FT=Feel/Think, JP=Judg/Perc';
COMMENT ON COLUMN preguntas.peso IS 'Peso en la fórmula: +1 o -1';

-- =====
-- TABLA: tipos_personalidad
-- =====

CREATE TABLE tipos_personalidad (
    codigo VARCHAR(4) PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    descripcion_corta TEXT NOT NULL,
    descripcion_completa TEXT NOT NULL,
    fortalezas TEXT NULL,
    debilidades TEXT NULL,
    carreras_sugeridas TEXT NULL,
    famosos_tipo TEXT NULL,
    porcentaje_poblacion DECIMAL(4,2) NULL,

    CONSTRAINT chk_codigo_formato CHECK (
        codigo ~ '^[IE][SN][FT][JP]'
    )
);

-- Comentarios de tabla tipos_personalidad
COMMENT ON TABLE tipos_personalidad IS 'Catálogo de los 16 tipos de personalidad MBTI';
COMMENT ON COLUMN tipos_personalidad.porcentaje_poblacion IS 'Porcentaje de la población mundial con este tipo';

-- =====

```

```

-- TABLA: sesiones_test
-- =====

CREATE TABLE sesiones_test (
    id_sesion SERIAL PRIMARY KEY,
    id_usuario INTEGER NULL,
    token_anonimo VARCHAR(64) UNIQUE NOT NULL,
    fecha_inicio TIMESTAMP NOT NULL DEFAULT NOW(),
    fecha_fin TIMESTAMP NULL,
    completado BOOLEAN NOT NULL DEFAULT FALSE,
    resultado_ie INTEGER NULL,
    resultado_sn INTEGER NULL,
    resultado_ft INTEGER NULL,
    resultado_jp INTEGER NULL,
    tipo_personalidad VARCHAR(4) NULL,
    ip_address VARCHAR(45) NULL,
    user_agent TEXT NULL,

    CONSTRAINT fk_sesiones_usuario
        FOREIGN KEY (id_usuario)
        REFERENCES usuarios(id_usuario)
        ON DELETE SET NULL,

    CONSTRAINT fk_sesiones_tipo
        FOREIGN KEY (tipo_personalidad)
        REFERENCES tipos_personalidad(codigo)
        ON DELETE SET NULL,

    CONSTRAINT chk_resultado_ie CHECK (resultado_ie IS NULL OR resultado_ie BETWEEN 8 AND 38),
    CONSTRAINT chk_resultado_sn CHECK (resultado_sn IS NULL OR resultado_sn BETWEEN 4 AND 44),
    CONSTRAINT chk_resultado_ft CHECK (resultado_ft IS NULL OR resultado_ft BETWEEN 8 AND 38),
    CONSTRAINT chk_resultado_jp CHECK (resultado_jp IS NULL OR resultado_jp BETWEEN 10 AND 40),
    CONSTRAINT chk_fecha_fin CHECK (fecha_fin IS NULL OR fecha_fin >= fecha_inicio),
    CONSTRAINT chk_completado_resultados CHECK (
        (completado = FALSE AND tipo_personalidad IS NULL) OR
        (completado = TRUE AND tipo_personalidad IS NOT NULL)
    )
);

-- Índices de sesiones_test
CREATE INDEX idx_sesiones_usuario ON sesiones_test(id_usuario);
CREATE INDEX idx_sesiones_token ON sesiones_test(token_anonimo);
CREATE INDEX idx_sesiones_tipo ON sesiones_test(tipo_personalidad);
CREATE INDEX idx_sesiones_fecha_inicio ON sesiones_test(fecha_inicio);
CREATE INDEX idx_sesiones_completado ON sesiones_test(completado);
CREATE INDEX idx_sesiones_usuario_completado ON sesiones_test(id_usuario, completado);

-- Comentarios de tabla sesiones_test

```

```
COMMENT ON TABLE sesiones_test IS 'Registra cada intento de test, anónimo o de usuario';
COMMENT ON COLUMN sesiones_test.token_anonimo IS 'Token único para sesiones anónimas, se mantiene tras registro';
COMMENT ON COLUMN sesiones_test.resultado_ie IS 'Puntuación IE: 8-38, punto de corte 24';
COMMENT ON COLUMN sesiones_test.resultado_sn IS 'Puntuación SN: 4-44, punto de corte 24';
COMMENT ON COLUMN sesiones_test.resultado_ft IS 'Puntuación FT: 8-38, punto de corte 24';
COMMENT ON COLUMN sesiones_test.resultado_jp IS 'Puntuación JP: 10-40, punto de corte 24';
```

```
-- =====
```

```
-- TABLA: respuestas
```

```
-- =====
```

```
CREATE TABLE respuestas (
    id_respuesta SERIAL PRIMARY KEY,
    id_sesion INTEGER NOT NULL,
    id_pregunta INTEGER NOT NULL,
    valor_respuesta INTEGER NOT NULL,
    tiempo_respuesta INTEGER NULL,
    fecha_respuesta TIMESTAMP NOT NULL DEFAULT NOW(),
```

```
CONSTRAINT fk_respuestas_sesion
    FOREIGN KEY (id_sesion)
        REFERENCES sesiones_test(id_sesion)
        ON DELETE CASCADE,
```

```
CONSTRAINT fk_respuestas_pregunta
    FOREIGN KEY (id_pregunta)
        REFERENCES preguntas(id_pregunta)
        ON DELETE RESTRICT,
```

```
CONSTRAINT chk_valor_respuesta CHECK (valor_respuesta BETWEEN 1 AND 5),
CONSTRAINT chk_tiempo_respuesta CHECK (tiempo_respuesta IS NULL OR tiempo_respuesta >= 0),
CONSTRAINT uk_sesion_pregunta UNIQUE (id_sesion, id_pregunta)
```

```
);
```

```
-- Índices de respuestas
```

```
CREATE INDEX idx_respuestas_sesion ON respuestas(id_sesion);
CREATE INDEX idx_respuestas_pregunta ON respuestas(id_pregunta);
CREATE INDEX idx_respuestas_fecha ON respuestas(fecha_respuesta);
```

```
-- Comentarios de tabla respuestas
```

```
COMMENT ON TABLE respuestas IS 'Almacena cada respuesta individual del test';
COMMENT ON COLUMN respuestas.valor_respuesta IS 'Valor de 1 a 5: 1=totalmente izquierda, 5=totalmente derecha';
COMMENT ON COLUMN respuestas.tiempo_respuesta IS 'Tiempo en segundos que tomó responder';
```

```
-- =====
```

```
-- TABLA: audit_log (Opcional - para auditoría)
```

```
-- =====
```

```
CREATE TABLE audit_log (
```

```
id_log SERIAL PRIMARY KEY,  
tabla_afectada VARCHAR(50) NOT NULL,  
operacion VARCHAR(10) NOT NULL,  
id_registro INTEGER NULL,  
usuario_sistema VARCHAR(100) NULL,  
fecha_operacion TIMESTAMP NOT NULL DEFAULT NOW(),  
datos_anteriores JSONB NULL,  
datos_nuevos JSONB NULL,  
ip_origen VARCHAR(45) NULL,  
  
CONSTRAINT chk_operacion CHECK (operacion IN ('INSERT', 'UPDATE', 'DELETE'))  
);
```

-- Índices de audit_log

```
CREATE INDEX idx_audit_tabla ON audit_log(tabla_afectada);  
CREATE INDEX idx_audit_fecha ON audit_log(fecha_operacion);  
CREATE INDEX idx_audit_operacion ON audit_log(operacion);
```

```
COMMENT ON TABLE audit_log IS 'Registro de auditoría de operaciones en la base de datos';
```

```
-- ======  
-- FIN DE CREACIÓN DE TABLAS  
-- ======
```

10.2 Script de Inserción de Datos Iniciales

sql

-- =====
-- DATOS INICIALES: PREGUNTAS DEL TEST OEJTS
-- =====

INSERT INTO preguntas (id_pregunta, texto_izquierda, texto_derecha, dimension, peso, orden) **VALUES**
(1, 'hace listas', 'confía en la memoria', 'JP', 1, 1),
(2, 'escéptico', 'quiere creer', 'FT', -1, 2),
(3, 'aburrido estando solo', 'necesita tiempo solo', 'IE', -1, 3),
(4, 'acepta las cosas como son', 'insatisfecho con cómo son las cosas', 'SN', 1, 4),
(5, 'mantiene un cuarto limpio', 'deja cosas donde sea', 'JP', 1, 5),
(6, 'piensa que "robótico" es un insulto', 'aspira a tener una mente mecánica', 'FT', 1, 6),
(7, 'energético', 'tranquilo', 'IE', -1, 7),
(8, 'prefiere examen de opción múltiple', 'prefiere respuestas de ensayo', 'SN', 1, 8),
(9, 'caótico', 'organizado', 'JP', -1, 9),
(10, 'se lastima fácilmente', 'piel gruesa', 'FT', 1, 10),
(11, 'trabaja mejor en grupos', 'trabaja mejor solo', 'IE', -1, 11),
(12, 'enfocado en el presente', 'enfocado en el futuro', 'SN', 1, 12),
(13, 'planea con mucha anticipación', 'planea a último minuto', 'JP', 1, 13),
(14, 'quiere el respeto de la gente', 'quiere su amor', 'FT', -1, 14),
(15, 'se agota en fiestas', 'se energiza en fiestas', 'IE', 1, 15),
(16, 'encaja', 'sobresale', 'SN', 1, 16),
(17, 'mantiene opciones abiertas', 'se compromete', 'JP', -1, 17),
(18, 'quiere ser bueno arreglando cosas', 'quiere ser bueno arreglando personas', 'FT', -1, 18),
(19, 'habla más', 'escucha más', 'IE', -1, 19),
(20, 'al describir un evento, dice qué pasó', 'al describir un evento, dice qué significó', 'SN', 1, 20),
(21, 'hace el trabajo de inmediato', 'procrastina', 'JP', 1, 21),
(22, 'sigue el corazón', 'sigue la cabeza', 'FT', 1, 22),
(23, 'se queda en casa', 'sale a la ciudad', 'IE', 1, 23),
(24, 'quiere la visión general', 'quiere los detalles', 'SN', -1, 24),
(25, 'improvista', 'prepara', 'JP', -1, 25),
(26, 'basa la moralidad en la justicia', 'basa la moralidad en la compasión', 'FT', -1, 26),
(27, 'le cuesta gritar muy fuerte', 'gritar a otros cuando están lejos viene naturalmente', 'IE', 1, 27),
(28, 'teórico', 'empírico', 'SN', -1, 28),
(29, 'trabaja duro', 'juega duro', 'JP', 1, 29),
(30, 'incómodo con las emociones', 'valora las emociones', 'FT', -1, 30),
(31, 'le gusta actuar frente a otras personas', 'evita hablar en público', 'IE', -1, 31),
(32, 'le gusta saber "quién", "qué", "cuándo"', 'le gusta saber "por qué"', 'SN', 1, 32);

-- =====
-- DATOS INICIALES: TIPOS DE PERSONALIDAD
-- =====

-- INTJ - El Arquitecto

INSERT INTO tipos_personalidad (codigo, nombre, descripcion_corta, descripcion_completa, fortalezas, debilidades, carrera
('INTJ', 'El Arquitecto', 'Estratega imaginativo con un plan para todo', 'Los INTJ son pensadores estratégicos que destacan por

('INTP', 'El Lógico', 'Innovador filosófico, sediento de conocimiento', 'Los INTP son pensadores analíticos que aman las teorías abstractas y complejas', 'Los INTP son visionarios y creativos')
('ENTJ', 'El Comandante', 'Líder audaz, imaginativo y de voluntad fuerte', 'Los ENTJ son líderes naturales que destacan en organizaciones y empresas')
('ENTP', 'El Innovador', 'Pensador inteligente y curioso que no puede resistir un desafío intelectual', 'Los ENTP son innovadores y creativos')
('INFJ', 'El Abogado', 'Idealista silencioso pero inspirador', 'Los INFJ son idealistas con profunda comprensión de las personas')
('INFP', 'El Mediador', 'Idealista poético, amable y altruista', 'Los INFP son soñadores guiados por valores profundos. Son creativos y empáticos')
('ENFJ', 'El Protagonista', 'Líder carismático e inspirador capaz de cautivar a su audiencia', 'Los ENFJ son líderes naturales motivadores')
('ENFP', 'El Activista', 'Espíritu libre entusiasta, creativo y sociable', 'Los ENFP son personas energéticas y creativas que ven el mundo como un escenario para su expresión')
('ISTJ', 'El Logista', 'Individuo práctico y enfocado en los hechos, cuya confiabilidad es incuestionable', 'Los ISTJ son trabajadores dedicados y responsables')
('ISFJ', 'El Defensor', 'Protector dedicado, dispuesto a defender a sus seres queridos', 'Los ISFJ son personas cálidas y consideradas')
('ESTJ', 'El Ejecutivo', 'Excelente administrador incomparable en gestionar cosas y personas', 'Los ESTJ son organizadores natos y eficientes')
('ESFJ', 'El Cónsul', 'Personas extraordinariamente cariñosas, sociables y populares', 'Los ESFJ son personas orientadas a ayudar y servir')
('ISTP', 'El Virtuoso', 'Experimentador audaz y práctico, maestro de todo tipo de herramientas', 'Los ISTP son solucionadores creativos')
('ISFP', 'El Aventurero', 'Artista flexible y encantador, siempre dispuesto a explorar y experimentar', 'Los ISFP son personas creativas y artísticas')
('ESTP', 'El Emprendedor', 'Personas inteligentes, enérgicas y muy perceptivas que disfrutan el riesgo', 'Los ESTP son personas dinámicas y vivas')
('ESFP', 'El Animador', 'Artistas espontáneos, enérgicos y entusiastas que nunca aburrirán a su audiencia', 'Los ESFP son personas divertidas y amigables')
-- ======
-- FIN DE DATOS INICIALES
-- ======

10.3 Funciones Almacenadas Útiles

sql

```

-- =====
-- FUNCIÓN: Calcular resultados del test
-- =====

CREATE OR REPLACE FUNCTION calcular_resultados_test(p_id_sesion INTEGER)
RETURNS TABLE (
    ie_score INTEGER,
    sn_score INTEGER,
    ft_score INTEGER,
    jp_score INTEGER,
    tipo VARCHAR(4)
) AS $$
DECLARE
    v_ie INTEGER := 30;
    v_sn INTEGER := 12;
    v_ft INTEGER := 30;
    v_jp INTEGER := 18;
    v_tipo VARCHAR(4);
    v_count INTEGER;
BEGIN
    -- Verificar que la sesión existe
    SELECT COUNT(*) INTO v_count
    FROM sesiones_test
    WHERE id_sesion = p_id_sesion;

    IF v_count = 0 THEN
        RAISE EXCEPTION 'Sesión no encontrada: %', p_id_sesion;
    END IF;

    -- Verificar que hay 32 respuestas
    SELECT COUNT(*) INTO v_count
    FROM respuestas
    WHERE id_sesion = p_id_sesion;

    IF v_count != 32 THEN
        RAISE EXCEPTION 'Sesión incompleta. Respuestas: %/32', v_count;
    END IF;

    -- Calcular IE
    SELECT v_ie -
        (SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 3) -
        (SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 7) -
        (SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 11) +
        (SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 15) -
        (SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 19) +
        (SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 23) +

```

```
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 27) -  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 31)  
INTO v_ie;
```

-- Calcular SN

```
SELECT v_sn +  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 4) +  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 8) +  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 12) +  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 16) +  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 20) -  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 24) -  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 28) +  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 32)  
INTO v_sn;
```

-- Calcular FT

```
SELECT v_ft -  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 2) +  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 6) +  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 10) -  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 14) -  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 18) +  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 22) -  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 26) -  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 30)  
INTO v_ft;
```

-- Calcular JP

```
SELECT v_jp +  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 1) +  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 5) -  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 9) +  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 13) -  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 17) +  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 21) -  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 25) +  
(SELECT valor_respuesta FROM respuestas WHERE id_sesion = p_id_sesion AND id_pregunta = 29)  
INTO v_jp;
```

-- Determinar tipo de personalidad

```
v_tipo := ";  
v_tipo := v_tipo || CASE WHEN v_ie > 24 THEN 'E' ELSE 'T' END;  
v_tipo := v_tipo || CASE WHEN v_sn > 24 THEN 'N' ELSE 'S' END;  
v_tipo := v_tipo || CASE WHEN v_ft > 24 THEN 'T' ELSE 'F' END;  
v_tipo := v_tipo || CASE WHEN v_jp > 24 THEN 'P' ELSE 'J' END;
```

```

-- Actualizar sesión con resultados
UPDATE sesiones_test
SET resultado_ie = v_ie,
    resultado_sn = v_sn,
    resultado_ft = v_ft,
    resultado_jp = v_jp,
    tipo_personalidad = v_tipo,
    completado = TRUE,
    fecha_fin = NOW()
WHERE id_sesion = p_id_sesion;

-- Retornar resultados
RETURN QUERY SELECT v_ie, v_sn, v_ft, v_jp, v_tipo;
END;
$ LANGUAGE plpgsql;

-- Comentario de función
COMMENT ON FUNCTION calcular_resultados_test IS 'Calcula los resultados del test OEJTS y actualiza la sesión';

-- =====
-- FUNCIÓN: Obtener perfil completo de usuario
-- =====

CREATE OR REPLACE FUNCTION obtener_perfil_usuario(p_id_sesion INTEGER)
RETURNS JSON AS $$
DECLARE
    v_resultado JSON;
BEGIN
    SELECT json_build_object(
        'sesion', json_build_object(
            'id_sesion', s.id_sesion,
            'fecha_inicio', s.fecha_inicio,
            'fecha_fin', s.fecha_fin,
            'tipo_personalidad', s.tipo_personalidad
        ),
        'puntuaciones', json_build_object(
            'IE', s.resultado_ie,
            'SN', s.resultado_sn,
            'FT', s.resultado_ft,
            'JP', s.resultado_jp
        ),
        'dimensiones', json_build_object(
            'extraversion_introversion', CASE WHEN s.resultado_ie > 24 THEN 'Extravertido' ELSE 'Introvertido' END,
            'intuicion_sensacion', CASE WHEN s.resultado_sn > 24 THEN 'Intuitivo' ELSE 'Sensorial' END,
            'pensamiento_sentimiento', CASE WHEN s.resultado_ft > 24 THEN 'Pensamiento' ELSE 'Sentimiento' END,
            'percepcion_juicio', CASE WHEN s.resultado_jp > 24 THEN 'Perceptivo' ELSE 'Juicioso' END
        )
    );

```

```

'tipo', json_build_object(
    'codigo', t.codigo,
    'nombre', t.nombre,
    'descripcion_corta', t.descripcion_corta,
    'descripcion_completa', t.descripcion_completa,
    'fortalezas', t.fortalezas,
    'debilidades', t.debilidades,
    'carreras_sugeridas', t.carreras_sugeridas,
    'famosos_tipo', t.famosos_tipo,
    'porcentaje_poblacion', t.porcentaje_poblacion
)
) INTO v_resultado
FROM sesiones_test s
LEFT JOIN tipos_personalidad t ON s.tipo_personalidad = t.codigo
WHERE s.id_sesion = p_id_sesion;

RETURN v_resultado;
END;
$ LANGUAGE plpgsql;

```

```
-- =====
-- FUNCIÓN: Validar sesión completa
-- =====
```

```

CREATE OR REPLACE FUNCTION validar_sesion_completa(p_id_sesion INTEGER)
RETURNS BOOLEAN AS $
DECLARE
    v_count INTEGER;
BEGIN
    SELECT COUNT(*)
    INTO v_count
    FROM respuestas
    WHERE id_sesion = p_id_sesion;

    RETURN v_count = 32;
END;
$ LANGUAGE plpgsql;

```

```
-- =====
-- FUNCIÓN: Generar token anónimo único
-- =====
```

```

CREATE OR REPLACE FUNCTION generar_token_anonimo()
RETURNS VARCHAR(64) AS $
DECLARE
    v_token VARCHAR(64);
    v_existe BOOLEAN;

```

```

BEGIN
  LOOP
    -- Generar token aleatorio
    v_token := encode(gen_random_bytes(32), 'hex');

    -- Verificar si ya existe
    SELECT EXISTS(
      SELECT 1 FROM sesiones_test WHERE token_anonimo = v_token
    ) INTO v_existe;

    EXIT WHEN NOT v_existe;
  END LOOP;

  RETURN v_token;
END;
$ LANGUAGE plpgsql;

-- =====
-- FUNCIÓN: Estadísticas generales del sistema
-- =====

CREATE OR REPLACE FUNCTION estadisticas_sistema()
RETURNS JSON AS $$
DECLARE
  v_resultado JSON;
BEGIN
  SELECT json_build_object(
    'total_usuarios', (SELECT COUNT(*) FROM usuarios WHERE activo = TRUE),
    'total_tests_completados', (SELECT COUNT(*) FROM sesiones_test WHERE completado = TRUE),
    'tests_hoy', (SELECT COUNT(*) FROM sesiones_test WHERE DATE(fecha_inicio) = CURRENT_DATE),
    'tipo_mas_comun', (
      SELECT tipo_personalidad
      FROM sesiones_test
      WHERE completado = TRUE AND tipo_personalidad IS NOT NULL
      GROUP BY tipo_personalidad
      ORDER BY COUNT(*) DESC
      LIMIT 1
    ),
    'distribucion_tipos', (
      SELECT json_object_agg(tipo_personalidad, cantidad)
      FROM (
        SELECT tipo_personalidad, COUNT(*) as cantidad
        FROM sesiones_test
        WHERE completado = TRUE AND tipo_personalidad IS NOT NULL
        GROUP BY tipo_personalidad
        ORDER BY tipo_personalidad
      ) sub
    )
  );
END;
$$ LANGUAGE plpgsql;

```

```

),
'promedio_tiempo_test', (
    SELECT AVG(EXTRACT(EPOCH FROM (fecha_fin - fecha_inicio))/60)::INTEGER
    FROM sesiones_test
    WHERE completado = TRUE AND fecha_fin IS NOT NULL
)
) INTO v_resultado;

RETURN v_resultado;
END;
$ LANGUAGE plpgsql;

-- =====
-- VISTAS ÚTILES
-- =====

-- Vista: Sesiones completas con información de usuario
CREATE OR REPLACE VIEW v_sesiones_completas AS
SELECT
    s.id_sesion,
    s.id_usuario,
    u.nombre_completo,
    u.email,
    s.fecha_inicio,
    s.fecha_fin,
    EXTRACT(EPOCH FROM (s.fecha_fin - s.fecha_inicio))/60 AS duracion_minutos,
    s.tipo_personalidad,
    s.resultado_ie,
    s.resultado_sn,
    s.resultado_ft,
    s.resultado_jp,
    t.nombre AS nombre_tipo,
    t.descripcion_corta
FROM sesiones_test s
LEFT JOIN usuarios u ON s.id_usuario = u.id_usuario
LEFT JOIN tipos_personalidad t ON s.tipo_personalidad = t.codigo
WHERE s.completado = TRUE;

COMMENT ON VIEW v_sesiones_completas IS 'Vista consolidada de sesiones completadas con datos de usuario y tipo';

-- Vista: Distribución de tipos de personalidad
CREATE OR REPLACE VIEW v_distribucion_tipos AS
SELECT
    t.codigo,
    t.nombre,
    COUNT(s.id_sesion) as cantidad_tests,
    ROUND(COUNT(s.id_sesion) * 100.0 / NULLIF(SUM(COUNT(s.id_sesion)) OVER(), 0), 2) as porcentaje_muestra,

```

```
t.porcentaje_poblacion as porcentaje_teorico
FROM tipos_personalidad t
LEFT JOIN sesiones_test s ON t.codigo = s.tipo_personalidad AND s.completado = TRUE
GROUP BY t.codigo, t.nombre, t.porcentaje_poblacion
ORDER BY cantidad_tests DESC;
```

```
COMMENT ON VIEW v_distribucion_tipos IS 'Distribución de tipos de personalidad en la muestra vs población general';
```

```
-- Vista: Estadísticas por usuario
```

```
CREATE OR REPLACE VIEW v_estadisticas_usuario AS
SELECT
    u.id_usuario,
    u.nombre_completo,
    u.email,
    COUNT(s.id_sesion) as tests_realizados,
    MAX(s.fecha_fin) as ultimo_test,
    MODE() WITHIN GROUP (ORDER BY s.tipo_personalidad) as tipo_mas_frecuente
FROM usuarios u
LEFT JOIN sesiones_test s ON u.id_usuario = s.id_usuario AND s.completado = TRUE
GROUP BY u.id_usuario, u.nombre_completo, u.email;
```

```
COMMENT ON VIEW v_estadisticas_usuario IS 'Estadísticas agregadas por usuario';
```

```
-- =====
-- TRIGGERS PARA AUDITORÍA
-- =====
```

```
-- Trigger: Registrar cambios en usuarios
```

```
CREATE OR REPLACE FUNCTION audit_usuarios()
RETURNS TRIGGER AS $
BEGIN
    IF TG_OP = 'INSERT' THEN
        INSERT INTO audit_log (tabla_afectada, operacion, id_registro, datos_nuevos)
        VALUES ('usuarios', 'INSERT', NEW.id_usuario, row_to_json(NEW));
    ELSIF TG_OP = 'UPDATE' THEN
        INSERT INTO audit_log (tabla_afectada, operacion, id_registro, datos_anteriores, datos_nuevos)
        VALUES ('usuarios', 'UPDATE', NEW.id_usuario, row_to_json(OLD), row_to_json(NEW));
    ELSIF TG_OP = 'DELETE' THEN
        INSERT INTO audit_log (tabla_afectada, operacion, id_registro, datos_anteriores)
        VALUES ('usuarios', 'DELETE', OLD.id_usuario, row_to_json(OLD));
    END IF;
    RETURN NULL;
END;
$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER trg_audit_usuarios
```

```
AFTER INSERT OR UPDATE OR DELETE ON usuarios
```

```
FOR EACH ROW EXECUTE FUNCTION audit_usuarios();
```

```
-- Trigger: Actualizar ultimo_acceso al hacer login
CREATE OR REPLACE FUNCTION actualizar_ultimo_acceso()
RETURNS TRIGGER AS $
BEGIN
    -- Este trigger se llamaría desde la aplicación al validar login
    RETURN NEW;
END;
$ LANGUAGE plpgsql;
```

```
-- =====
-- CONSULTAS ÚTILES PARA LA APLICACIÓN
-- =====
```

```
-- Consulta 1: Obtener todas las preguntas en orden
-- SELECT * FROM preguntas WHERE activa = TRUE ORDER BY orden;

-- Consulta 2: Crear nueva sesión anónima
-- INSERT INTO sesiones_test (token_anonimo, ip_address, user_agent)
-- VALUES (generar_token_anonimo(), '192.168.1.1', 'Mozilla/5.0...')
-- RETURNING id_sesion, token_anonimo;

-- Consulta 3: Guardar respuesta
-- INSERT INTO respuestas (id_sesion, id_pregunta, valor_respuesta, tiempo_respuesta)
-- VALUES (1, 1, 3, 5)
-- ON CONFLICT (id_sesion, id_pregunta) DO UPDATE
-- SET valor_respuesta = EXCLUDED.valor_respuesta;

-- Consulta 4: Asociar sesión a usuario tras registro
-- UPDATE sesiones_test
-- SET id_usuario = 123
-- WHERE token_anonimo = 'abc...xyz';

-- Consulta 5: Calcular y obtener resultados
-- SELECT * FROM calcular_resultados_test(1);

-- Consulta 6: Obtener perfil completo
-- SELECT * FROM obtener_perfil_usuario(1);

-- Consulta 7: Historial de tests de un usuario
-- SELECT * FROM v_sesiones_completas
-- WHERE id_usuario = 123
-- ORDER BY fecha_inicio DESC;

-- Consulta 8: Verificar si sesión está completa
-- SELECT validar_sesion_completa(1);
```

-- =====

-- FIN DEL SCRIPT

-- =====