FIAP

TECNÓLOGO EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

Gabriel Lima Silva - RM 556639

Cauã Marcelo Da Silva Machado - RM 558024

Marcos Ramalho - RM 554611

MASTERING RELATIONAL AND NON-RELATIONAL DATABASE

São Paulo – SP

2025

FIAP

TECNÓLOGO EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

Gabriel Lima Silva - RM 556639

Cauã Marcelo Da Silva Machado - RM 558024

Marcos Ramalho - RM 554611

MASTERING RELATIONAL AND NON-RELATIONAL DATABASE

Trabalho apresentado à disciplina Masterning Relational And Non-Relational Database Curso Tecnólogo em Análise e Desenvolvimento de Sistemas da FIAP - Campus Paulista, como requisito parcial para a obtenção de nota.

Professor: Vergílio Valério dos Santos

São Paulo – SP

2025

SUMARIO

HEALTHHELP

Resumo Executivo

O HealthHelp é uma solução tecnológica desenvolvida para promover o equilíbrio da rotina e o bem-estar físico e mental dos usuários. O projeto integra banco de dados relacional Oracle, modelagem NoSQL com MongoDB e geração automatizada de JSON para consumo por sistemas de Inteligência Artificial. A plataforma avalia hábitos diários, calcula um score de equilíbrio e recomenda ajustes personalizados ao usuário. O objetivo é fornecer uma ferramenta moderna, inteligente e acessível para gerenciamento de rotina, alinhada às exigências do mercado digital e às tendências do futuro do trabalho.

2. Descrição do Projeto

O HealthHelp é um sistema que coleta dados da rotina dos usuários (sono, trabalho, exercícios, lazer, transporte etc.), processa essas informações por meio de funções e procedures no Oracle, gera JSON estruturado, envia os dados para um banco NoSQL e oferece recomendações personalizadas. O sistema inclui auditoria completa, análise de pontuação e exportação automatizada de datasets.

3. Análise de Mercado

O mercado de aplicativos de autocuidado e produtividade cresce continuamente. A demanda por soluções que unam saúde, rotina, equilíbrio emocional e organização pessoal vem aumentando com a expansão do trabalho remoto e da necessidade de gestão inteligente do tempo. Empresas como Calm, Headspace e Google Fit mostram a força desse setor, mas ainda existe espaço para soluções focadas em “equilíbrio de rotina completo”.

4. Análise da Concorrência

Entre os principais concorrentes estão:

- Google Fit: monitora atividades físicas, mas não analisa rotina completa.
- Calm / Headspace: focados em saúde mental, com pouca integração à rotina diária.
- HabitNow / Notion: ótimos organizadores, mas não possuem análise de equilíbrio.

O HealthHelp se diferencia por integrar dados reais, auditoria, inteligência artificial, avaliação automatizada, exportação JSON e conexão híbrida Oracle + MongoDB.

5. Proposta de Valor

- Avaliação inteligente da rotina do usuário.
- Recomendação personalizada baseada em análise de dados.
- Estrutura híbrida Relacional + NoSQL.
- Funcionalidades preparadas para IA.
- Monitoramento de score diário.
- Transparência por meio de auditoria integrada.

6. Estratégia de Marketing

- Divulgação digital em redes sociais.
- Parcerias com academias, nutricionistas e psicólogos.
- Pacote premium com insights avançados baseados em IA.
- Integração com Apple Health e Google Fit.
- Produção de conteúdo educacional sobre bem-estar.

7. Operações

- Banco Oracle controlando integridade relacional e auditoria.
- Procedures centralizando regras de negócio.
- Exportação automática de JSON para análise externa.
- MongoDB armazenando dados consolidados e otimizados para IA.
- Potencial integração com APIs externas.

8. Plano Financeiro

Investimentos principais:

- Infraestrutura de banco de dados.
- Hospedagem do backend da aplicação.
- Custos com APIs externas, se integradas.
- Marketing digital.

Possíveis receitas:

- Assinatura mensal.
- Modelos freemium.
- Parcerias com clínicas e academias.

9. Equipe

- Banco de Dados: desenvolvimento das tabelas, procedures, triggers e auditoria.
- Backend/Java: integração com Oracle e MongoDB.
- Mobile/Frontend: interface de uso do app.
- Data Science/IA: processamento de JSONs e análise.
- Gestão/Documentação: organização, vídeo e apresentação.

10. Precificação

Modelo recomendado:

- Plano gratuito com recursos limitados.
- Plano Plus com exportação de relatórios e IA básica.
- Plano Premium com IA avançada e dashboards completos.

11. Acordo de Nível de Serviço (SLA)

- Disponibilidade do serviço: 99%.
- Tempo máximo de resposta da API: 500 ms.
- Atualizações semanais.
- Suporte via e-mail/WhatsApp com atendimento em até 12h.
- Auditoria permanente via triggers no Oracle.

12. Funcionalidades

Oracle:

- CRUD completo.
- Auditoria automática.
- Geração manual de JSON.
- Cálculo de score.
- Exportação completa de dataset.
- 450 registros diários e 2250 atividades geradas automaticamente.

MongoDB:

Estrutura NoSQL otimizada.

- Coleção `usuarios_rotina` com 30 documentos.

Índices:

- Email (unique)
- Score (desc)

Consultas avançadas:

- Top 5 scores
- Scores críticos
- Média por gênero
- Bucket de scores

App/Backend:

- Avaliação automática da rotina.
- Recomendações personalizadas.
- Suporte futuro para IA generativa.

13. Tecnologias Utilizadas

- Oracle Database
- PL/SQL (Triggers, Procedures, Packages)
- MongoDB Compass + Mongo Shell
- MongoImport / NoSQL Modelling
- JSON Manual
- Java (camada de integração)
- Ferramentas da FIAP para modelagem

14. Referências

- Documentação Oracle
- Documentação MongoDB
- Papers e artigos sobre bem-estar e equilíbrio de rotina
- Materiais das aulas FIAP
- OWASP (para boas práticas de segurança)

15. Conclusão

O HealthHelp une tecnologia, análise de dados e bem-estar em uma solução completa, robusta e escalável. O sistema foi projetado com rigor técnico e compatibilidade com ambientes corporativos modernos. A integração entre Oracle e MongoDB permite processamento estruturado e análise flexível, tornando o projeto totalmente alinhado ao futuro do trabalho e ao uso de IA para promover saúde e equilíbrio na rotina de seus usuários.

16. Script SQL do Banco de Dados Relacional

Nesta seção será apresentado o script completo utilizado para criação e carga do banco de dados relacional Oracle do projeto HealthHelp, incluindo:

- Criação das tabelas:
  `USUARIO`, `CATEGORIA_ATIVIDADE`, `REGISTRO_DIARIO`, `ATIVIDADE`, `HABITO`, `RECOMENDACAO`, `AUDIT_LOG`.
- Definição de chaves primárias, estrangeiras e restrições de unicidade.
- Criação das triggers de auditoria:
  `TRG_AUDIT_USUARIO`, `TRG_AUDIT_REGISTRO`, `TRG_AUDIT_ATIVIDADE`, `TRG_AUDIT_RECOMENDACAO`.
- Criação do package `PKG_WELLNESS` (spec e body).
- Procedures de carga automática de dados (30 usuários, 450 registros diários, atividades, hábitos e recomendações).
- Blocos de teste para geração de JSON e cálculo de score.

Espaço reservado para inserção do script completo em anexo pelo grupo.

-- 01) DROP SEGURO – LIMPEZA COMPLETA DO SCHEMA

BEGIN FOR t IN (SELECT object_name, object_type FROM user_objects WHERE object_type IN ('TABLE','TRIGGER','PACKAGE','PACKAGE BODY') AND object_name IN ('USUARIO','CATEGORIA_ATIVIDADE','REGISTRO_DIARIO','ATIVIDADE', 'HABITO','RECOMENDACAO','AUDIT_LOG', 'TRG_AUDIT_USUARIO','TRG_AUDIT_REGISTRO', 'TRG_AUDIT_ATIVIDADE','TRG_AUDIT_RECOMENDACAO', 'PKG_WELLNESS')) LOOP BEGIN EXECUTE IMMEDIATE 'DROP '||t.object_type||' '||t.object_name||' CASCADE CONSTRAINTS'; EXCEPTION WHEN OTHERS THEN NULL; END; END LOOP; END; /

-- 02) CRIAÇÃO DAS TABELAS HEALTHHELP

CREATE TABLE usuario(usuario_id NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY, nome VARCHAR2(100) NOT NULL, email VARCHAR2(200) NOT NULL UNIQUE, genero CHAR(1) CHECK (genero IN ('M','F')), dt_nascimento DATE, altura_cm NUMBER(5,2), peso_kg NUMBER(6,2), dt_cadastro DATE DEFAULT SYSDATE); /

CREATE TABLE categoria_atividade(categoria_id NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY, nome_categoria VARCHAR2(60) NOT NULL UNIQUE); /

CREATE TABLE registro_diario(registro_id NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY, usuario_id NUMBER NOT NULL REFERENCES usuario(usuario_id), data_ref DATE NOT NULL, pontuacao_equilibrio NUMBER(5,2), CONSTRAINT uq_reg UNIQUE(usuario_id, data_ref)); /

CREATE TABLE atividade(atividade_id NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY, registro_id NUMBER NOT NULL REFERENCES registro_diario(registro_id), categoria_id NUMBER NOT NULL REFERENCES categoria_atividade(categoria_id), descricao VARCHAR2(200), inicio_ts TIMESTAMP, fim_ts TIMESTAMP, intensidade_1a5 NUMBER(1), qualidade_1a5 NUMBER(1)); /

CREATE TABLE habito(habito_id NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY, usuario_id NUMBER NOT NULL REFERENCES usuario(usuario_id), categoria_id NUMBER NOT NULL REFERENCES categoria_atividade(categoria_id), nome VARCHAR2(100), objetivo_min_dia NUMBER(6)); /

CREATE TABLE recomendacao(recomendacao_id NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY, usuario_id NUMBER NOT NULL

REFERENCES usuario(usuario_id), data_ref DATE NOT NULL, texto VARCHAR2(1000), origem VARCHAR2(50), score_relevancia NUMBER(5,2)); /

CREATE TABLE audit_log(audit_id NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY, quando_ts TIMESTAMP DEFAULT SYSTIMESTAMP, tabela VARCHAR2(40), operacao VARCHAR2(10), chave VARCHAR2(100), detalhes VARCHAR2(4000)); /

-- 03) TRIGGERS DE AUDITORIA

CREATE OR REPLACE TRIGGER trg_audit_usuario AFTER INSERT OR UPDATE OR DELETE ON usuario FOR EACH ROW DECLARE v_op VARCHAR2(10); BEGIN IF INSERTING THEN v_op:='INSERT'; ELSIF UPDATING THEN v_op:='UPDATE'; ELSE v_op:='DELETE'; END IF;

INSERT INTO audit_log(tabela,operacao,chave,detalhes) VALUES('USUARIO',v_op, COALESCE(:NEW.usuario_id,:OLD.usuario_id), 'email='||COALESCE(:NEW.email,:OLD.email)); END; / CREATE OR REPLACE TRIGGER trg_audit_registro AFTER INSERT OR UPDATE OR DELETE ON registro_diario FOR EACH ROW DECLARE v_op VARCHAR2(10); BEGIN IF INSERTING THEN v_op:='INSERT'; ELSIF UPDATING THEN v_op:='UPDATE'; ELSE v_op:='DELETE'; END IF;

INSERT INTO audit_log(tabela,operacao,chave,detalhes) VALUES('REGISTRO_DIARIO',v_op, COALESCE(:NEW.registro_id,:OLD.registro_id), 'usuario='||COALESCE(:NEW.usuario_id,:OLD.usuario_id)); END; / CREATE OR REPLACE TRIGGER trg_audit_atividade AFTER INSERT OR UPDATE OR DELETE ON atividade FOR EACH ROW DECLARE v_op VARCHAR2(10); BEGIN IF INSERTING THEN v_op:='INSERT'; ELSIF UPDATING THEN v_op:='UPDATE'; ELSE v_op:='DELETE'; END IF;

INSERT INTO audit_log(tabela,operacao,chave,detalhes) VALUES('ATIVIDADE',v_op, COALESCE(:NEW.atividade_id,:OLD.atividade_id),

'registro='||COALESCE(:NEW.registro_id,:OLD.registro_id)); END; / CREATE OR REPLACE TRIGGER trg_audit_recomendacao AFTER INSERT OR UPDATE OR DELETE ON recomendacao FOR EACH ROW DECLARE v_op VARCHAR2(10); BEGIN IF INSERTING THEN v_op:='INSERT'; ELSIF UPDATING THEN v_op:='UPDATE'; ELSE v_op:='DELETE'; END IF;

INSERT INTO audit_log(tabela,operacao,chave,detalhes) VALUES('RECOMENDACAO',v_op, COALESCE(:NEW.recomendacao_id,:OLD.recomendacao_id), 'usuario='||COALESCE(:NEW.usuario_id,:OLD.usuario_id)); END; /

-- 04) PACKAGE SPEC

CREATE OR REPLACE PACKAGE pkg_wellness AS FUNCTION fn_validar_email(p_email VARCHAR2) RETURN NUMBER; FUNCTION fn_calc_score(p_registro_id NUMBER) RETURN NUMBER; FUNCTION fn_gerar_json_rotina(p_usuario_id NUMBER, p_data DATE) RETURN CLOB;

PROCEDURE prc_inserir_usuario(p_nome VARCHAR2, p_email VARCHAR2, p_dt_nasc DATE, p_altura NUMBER, p_peso NUMBER, p_genero VARCHAR2, p_usuario_id OUT NUMBER);

PROCEDURE prc_export_json_usuario(p_usuario_id NUMBER, p_json OUT CLOB); END pkg_wellness; /

-- 05) PACKAGE BODY (CORRIGIDO)

CREATE OR REPLACE PACKAGE BODY pkg_wellness AS

FUNCTION fn_validar_email(p_email VARCHAR2) RETURN NUMBER IS BEGIN IF REGEXP_LIKE(p_email,'^[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+.[A-Za-z]{2,}$') THEN RETURN 1; ELSE RETURN 0; END IF; END;

FUNCTION fn_calc_score(p_registro_id NUMBER) RETURN NUMBER IS v1 NUMBER; v2 NUMBER; BEGIN SELECT NVL(AVG(intensidade_1a5),0), NVL(AVG(qualidade_1a5),0) INTO v1, v2 FROM atividade WHERE registro_id = p_registro_id;

```
    RETURN      ROUND((v1*6      +      v2*4)      *      2,      2);
```

END;

FUNCTION fn_gerar_json_rotina(p_usuario_id NUMBER, p_data DATE) RETURN CLOB IS v_reg NUMBER; v_nome VARCHAR2(100); v_score NUMBER := 0; v_json CLOB; BEGIN SELECT nome INTO v_nome FROM usuario WHERE usuario_id = p_usuario_id;

```
    BEGIN
  SELECT             registro_id             INTO             v_reg
  FROM                                              registro_diario
  WHERE            usuario_id            =            p_usuario_id
    AND            data_ref            =            TRUNC(p_data);

  v_score                 :=                fn_calc_score(v_reg);

  v_json                           :=                           '{'||
    '"usuario_id":'||p_usuario_id||','||
    '"nome":"'                   ||REPLACE(v_nome,'"','\"')||'",'||
    '"data":"'        ||TO_CHAR(TRUNC(p_data),'YYYY-MM-DD')||'",'||
    '"score_rotina":'||v_score||
  '}';

  RETURN                                                    v_json;
```

```
EXCEPTION WHEN NO_DATA_FOUND THEN
  RETURN '{"erro":"registro nao encontrado"}';
END;
```

END;

PROCEDURE prc_inserir_usuario(p_nome VARCHAR2, p_email VARCHAR2, p_dt_nasc DATE, p_altura NUMBER, p_peso NUMBER, p_genero VARCHAR2, p_usuario_id OUT NUMBER) IS BEGIN IF fn_validar_email(p_email)=0 THEN RAISE_APPLICATION_ERROR(-20010,'Email inválido'); END IF;

```
INSERT INTO usuario(nome,email,dt_nascimento,genero,altura_cm,peso_kg)
VALUES (p_nome,p_email,p_dt_nasc,p_genero,p_altura,p_peso)
RETURNING usuario_id INTO p_usuario_id;
```

END;

PROCEDURE prc_export_json_usuario(p_usuario_id NUMBER, p_json OUT CLOB) IS v_nome VARCHAR2(100); BEGIN p_json := '{';

```
SELECT nome INTO v_nome FROM usuario WHERE usuario_id = p_usuario_id;

p_json := p_json||'"usuario_id":'||p_usuario_id||',';
p_json := p_json||'"nome":"' ||REPLACE(v_nome,'"','\"')||'",';

-------------------------------------------------------------
--                                                      ROTINA
-------------------------------------------------------------
p_json := p_json||'"rotina":[';
```

```
FOR r IN (
  SELECT registro_id,data_ref,fn_calc_score(registro_id) AS score
  FROM registro_diario
  WHERE usuario_id = p_usuario_id
  ORDER BY data_ref DESC
) LOOP
  p_json := p_json||
  '{'||'"data":"' ||TO_CHAR(r.data_ref,'YYYY-MM-DD')||'",'||
  '"score":'||r.score||','||
  '"atividades":[';

  FOR a IN (
    SELECT categoria_id,descricao,intensidade_1a5,qualidade_1a5
    FROM atividade
    WHERE registro_id = r.registro_id
  ) LOOP
    p_json := p_json||
    '{'||'"categoria_id":'||a.categoria_id||','||
    '"descricao":"' ||REPLACE(a.descricao,'"','\"')||'",'||
    '"intensidade":'||a.intensidade_1a5||','||
    '"qualidade":'||a.qualidade_1a5||'},';
  END LOOP;

  IF SUBSTR(p_json,-1) = ',' THEN
    p_json := SUBSTR(p_json,1,LENGTH(p_json)-1);
  END IF;

  p_json := p_json||']},';
END LOOP;

IF SUBSTR(p_json,-1) = ',' THEN
  p_json := SUBSTR(p_json,1,LENGTH(p_json)-1);
```

```
END                                                                IF;

p_json                        :=                        p_json||'],';

-------------------------------------------------------------
--                                                          HÁBITOS
-------------------------------------------------------------
p_json                    :=                    p_json||'"habitos":[';

FOR                     h                     IN                     (
  SELECT                          categoria_id,nome,objetivo_min_dia
  FROM                                                          habito
  WHERE             usuario_id             =             p_usuario_id
)                                                                  LOOP
  p_json                          :=                          p_json||
  '{'||'"categoria_id":'||h.categoria_id||','||
  '"nome":"'                      ||REPLACE(h.nome,'"','\"')||'",'||
  '"objetivo":'||h.objetivo_min_dia||'},';
END                                                               LOOP;

IF          SUBSTR(p_json,-1)          =          ','          THEN
  p_json            :=            SUBSTR(p_json,1,LENGTH(p_json)-1);
END                                                                IF;

p_json                        :=                        p_json||'],';

-------------------------------------------------------------
--                                                      RECOMENDAÇÕES
-------------------------------------------------------------
p_json                :=                p_json||'"recomendacoes":[';

FOR                     rec                     IN                     (
```

```
  SELECT                    data_ref,texto,origem,score_relevancia
  FROM                                                 recomendacao
  WHERE            usuario_id            =            p_usuario_id
)                                                              LOOP
  p_json                         :=                       p_json||
  '{'||'"data":"'      ||TO_CHAR(rec.data_ref,'YYYY-MM-DD')||'",'||
  '"texto":"'                ||REPLACE(rec.texto,'"','\"')||'",'||
  '"origem":"'                                  ||rec.origem||'",'||
  '"score":'||rec.score_relevancia||'},';
END                                                            LOOP;

IF          SUBSTR(p_json,-1)          =          ','          THEN
  p_json           :=           SUBSTR(p_json,1,LENGTH(p_json)-1);
END                                                             IF;

p_json                         :=                       p_json||']';

p_json                         :=                       p_json||'}';
```

END;

END pkg_wellness; /

-- 06) CARGA MANUAL – 15 CATEGORIAS

INSERT INTO categoria_atividade(nome_categoria) VALUES ('SONO'); INSERT INTO categoria_atividade(nome_categoria) VALUES ('TRABALHO'); INSERT INTO categoria_atividade(nome_categoria) VALUES ('EXERCICIO'); INSERT INTO categoria_atividade(nome_categoria) VALUES ('ALIMENTACAO'); INSERT INTO categoria_atividade(nome_categoria) VALUES ('ESTUDO'); INSERT INTO categoria_atividade(nome_categoria) VALUES ('MEDITACAO'); INSERT INTO

categoria_atividade(nome_categoria) VALUES ('HIDRATACAO'); INSERT INTO categoria_atividade(nome_categoria) VALUES ('LAZER'); INSERT INTO categoria_atividade(nome_categoria) VALUES ('SOCIAL'); INSERT INTO categoria_atividade(nome_categoria) VALUES ('MUSICA'); INSERT INTO categoria_atividade(nome_categoria) VALUES ('TRANSPORTE'); INSERT INTO categoria_atividade(nome_categoria) VALUES ('RELAXAMENTO'); INSERT INTO categoria_atividade(nome_categoria) VALUES ('PLANEJAMENTO'); INSERT INTO categoria_atividade(nome_categoria) VALUES ('HOBBY'); INSERT INTO categoria_atividade(nome_categoria) VALUES ('LEITURA'); COMMIT;

-- 07) CARGA AUTOMÁTICA – 30 USUÁRIOS

BEGIN FOR i IN 1..30 LOOP INSERT INTO usuario(nome,email,dt_nascimento,genero,altura_cm,peso_kg) VALUES('Usuario '||i, 'usuario'||LPAD(i,2,'0')||'@healthhelp.com', ADD_MONTHS(DATE '1990-01-01',-i*10), CASE WHEN MOD(i,2)=0 THEN 'F' ELSE 'M' END, 160+MOD(i,15), 60+MOD(i,20)); END LOOP; COMMIT; END; /

-- 08) CARGA – 15 REGISTROS POR USUÁRIO (450 TOTAL)

BEGIN FOR u IN (SELECT usuario_id FROM usuario) LOOP FOR d IN 1..15 LOOP INSERT INTO registro_diario(usuario_id,data_ref,pontuacao_equilibrio) VALUES(u.usuario_id,TRUNC(SYSDATE)-d,ROUND(DBMS_RANDOM.VALUE(40,90),2)); END LOOP; END LOOP; COMMIT; END; /

-- 09) CARGA – ATIVIDADES (2250 TOTAL)

BEGIN FOR r IN (SELECT registro_id,data_ref FROM registro_diario) LOOP FOR c IN (SELECT categoria_id FROM categoria_atividade WHERE categoria_id <= 5) LOOP INSERT INTO atividade(registro_id,categoria_id,descricao,inicio_ts,fim_ts,intensidade_1a5,qualidade_1a5) VALUES(r.registro_id, c.categoria_id, 'Atividade rotineira', r.data_ref + (c.categoria_id+7)/24, r.data_ref + (c.categoria_id+8)/24, TRUNC(DBMS_RANDOM.VALUE(1,5)), TRUNC(DBMS_RANDOM.VALUE(1,5))); END LOOP; END LOOP; COMMIT; END; /

-- 10) CARGA – HÁBITOS (30 TOTAL)

BEGIN FOR u IN (SELECT usuario_id FROM usuario) LOOP INSERT INTO habito(usuario_id,categoria_id,nome,objetivo_min_dia) VALUES(u.usuario_id,3,'Exercicio Diário',30); END LOOP; COMMIT; END; /

-- 11) CARGA – RECOMENDAÇÕES (30 TOTAL)

BEGIN FOR u IN (SELECT usuario_id FROM usuario) LOOP INSERT INTO recomendacao(usuario_id,data_ref,texto,origem,score_relevancia) VALUES(u.usuario_id, TRUNC(SYSDATE), 'Sugestão gerada automaticamente', 'AI', ROUND(DBMS_RANDOM.VALUE(70,100),2)); END LOOP; COMMIT; END; /

-- 12) PROCEDURE – EXPORTAÇÃO DO DATASET COMPLETO PARA JSON (MONGO/IA)

CREATE OR REPLACE PROCEDURE prc_export_dataset_json(p_json OUT CLOB) IS v_media_txt VARCHAR2(50); BEGIN p_json := '{"usuarios":[';

FOR u IN (SELECT u.usuario_id, u.nome, u.email, u.genero, u.dt_nascimento, u.altura_cm, u.peso_kg, NVL(ROUND(AVG(r.pontuacao_equilibrio),2),0) AS media_score FROM usuario u LEFT JOIN registro_diario r ON r.usuario_id = u.usuario_id GROUP BY u.usuario_id,u.nome,u.email,u.genero, u.dt_nascimento,u.altura_cm,u.peso_kg ORDER BY u.usuario_id) LOOP

```
    v_media_txt                    :=                    TO_CHAR(
  u.media_score,
  'FM9990D00',
  'NLS_NUMERIC_CHARACTERS=.,'
);

p_json                         :=                         p_json||
  '{'||
  '"usuario_id":'||u.usuario_id||','||
  '"nome":"'                     ||REPLACE(u.nome,'"','\"')||'",'||
  '"email":"'                   ||REPLACE(u.email,'"','\"')||'",'||
  '"genero":"'                                    ||u.genero||'",'||
  '"dt_nascimento":"'           ||TO_CHAR(u.dt_nascimento,'YYYY-MM-
DD')||'",'||
  '"altura_cm":'||NVL(u.altura_cm,0)||','||
  '"peso_kg":'||NVL(u.peso_kg,0)||','||
  '"media_score_rotina":'||v_media_txt||
  '},';
```

END LOOP;

IF SUBSTR(p_json,-1) = ',' THEN p_json := SUBSTR(p_json,1,LENGTH(p_json)-1); END IF;

p_json := p_json||']}'; END; /

-- 13) TESTE – GERAR JSON INDIVIDUAL (ROTINA DE UM DIA)

SET SERVEROUTPUT ON DECLARE v_json CLOB; BEGIN v_json := pkg_wellness.fn_gerar_json_rotina(1,TRUNC(SYSDATE)-1); DBMS_OUTPUT.PUT_LINE(v_json); END; /

-- 14) TESTE – EXPORTAÇÃO DO DATASET COMPLETO (PARA ARQUIVO .JSON)

SET SERVEROUTPUT ON DECLARE v_json CLOB; BEGIN prc_export_dataset_json(v_json); DBMS_OUTPUT.PUT_LINE(v_json); END; /

-- 15) VALIDAÇÃO FINAL – CONTAGEM DOS REGISTROS

SET SERVEROUTPUT ON DECLARE v1 NUMBER;v2 NUMBER;v3 NUMBER;v4 NUMBER;v5 NUMBER;v6 NUMBER;v7 NUMBER; BEGIN SELECT COUNT*() INTO v1 FROM usuario; SELECT COUNT*()* INTO v2 FROM categoria_atividade; SELECT COUNT*() INTO v3 FROM registro_diario; SELECT COUNT*()* INTO v4 FROM atividade; SELECT COUNT*() INTO v5 FROM habito; SELECT COUNT*()* INTO v6 FROM recomendacao; SELECT COUNT(*) INTO v7 FROM audit_log;

DBMS_OUTPUT.PUT_LINE('usuarios.....: '||v1); DBMS_OUTPUT.PUT_LINE('categoria_atividade..: '||v2);

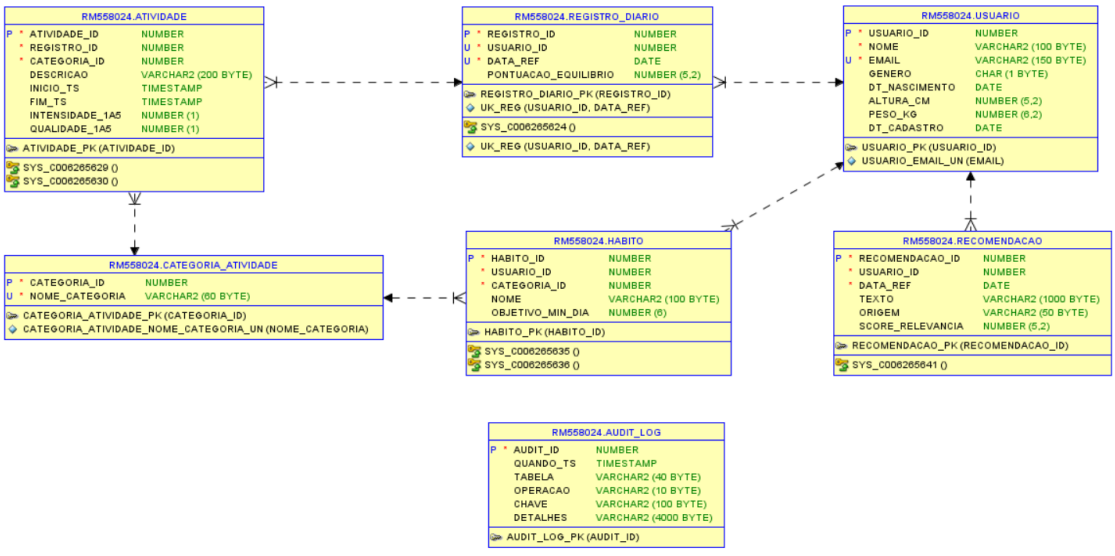
```
DBMS_OUTPUT.PUT_LINE('registro_diario......:                    '||v3);
DBMS_OUTPUT.PUT_LINE('atividade.....:                          '||v4);
DBMS_OUTPUT.PUT_LINE('habito.....:                             '||v5);
DBMS_OUTPUT.PUT_LINE('recomendacao.....:                       '||v6);
DBMS_OUTPUT.PUT_LINE('audit_log.....: '||v7); END; /
```

17. Evidências de Execução e Testes

Esta seção será utilizada para registrar as evidências práticas da execução do projeto, incluindo:

- Prints da criação das tabelas no Oracle.
- Prints da compilação das triggers e packages.
- Prints da execução das procedures de carga.
- Saída dos `SELECT COUNT` em todas as tabelas (confirmando quantidades).
- Saída do JSON gerado pelo Oracle (rotina individual e dataset consolidado).
- Prints do MongoDB Compass mostrando:
  - Banco `healthhelp_db`
  - Coleção `usuarios_rotina`
  - Índices criados (email e score)
  - Execução das consultas e aggregations.

Espaço reservado para inserção de imagens, prints e resultados pelo grupo.



```
-- 01) DROP SEGURO - LIMPEZA COMPLETA DO SCHEMA
-- ------------------------------------------------------------------------
BEGIN
  FOR t IN (
    SELECT object_name, object_type
    FROM user_objects
    WHERE object_type IN ('TABLE','TRIGGER','PACKAGE','PACKAGE BODY')
      AND object_name IN (
        'USUARIO','CATEGORIA_ATIVIDADE','REGISTRO_DIARIO','ATIVIDADE',
        'HABITO','RECOMENDACAO','AUDIT_LOG',
        'TRG_AUDIT_USUARIO','TRG_AUDIT_REGISTRO',
        'TRG_AUDIT_ATIVIDADE','TRG_AUDIT_RECOMENDACAO',
        'PKG_WELLNESS'
      )
  ) LOOP
    BEGIN
      EXECUTE IMMEDIATE 'DROP '||t.object_type||' '||t.object_name||' CASCADE CONSTRAINTS';
    EXCEPTION WHEN OTHERS THEN NULL;
    END;
  END LOOP;
END;
/
```

Saída do Script

Tarefa concluída em 10,658 segundos

```
Procedimento PL/SQL concluído com sucesso.
```

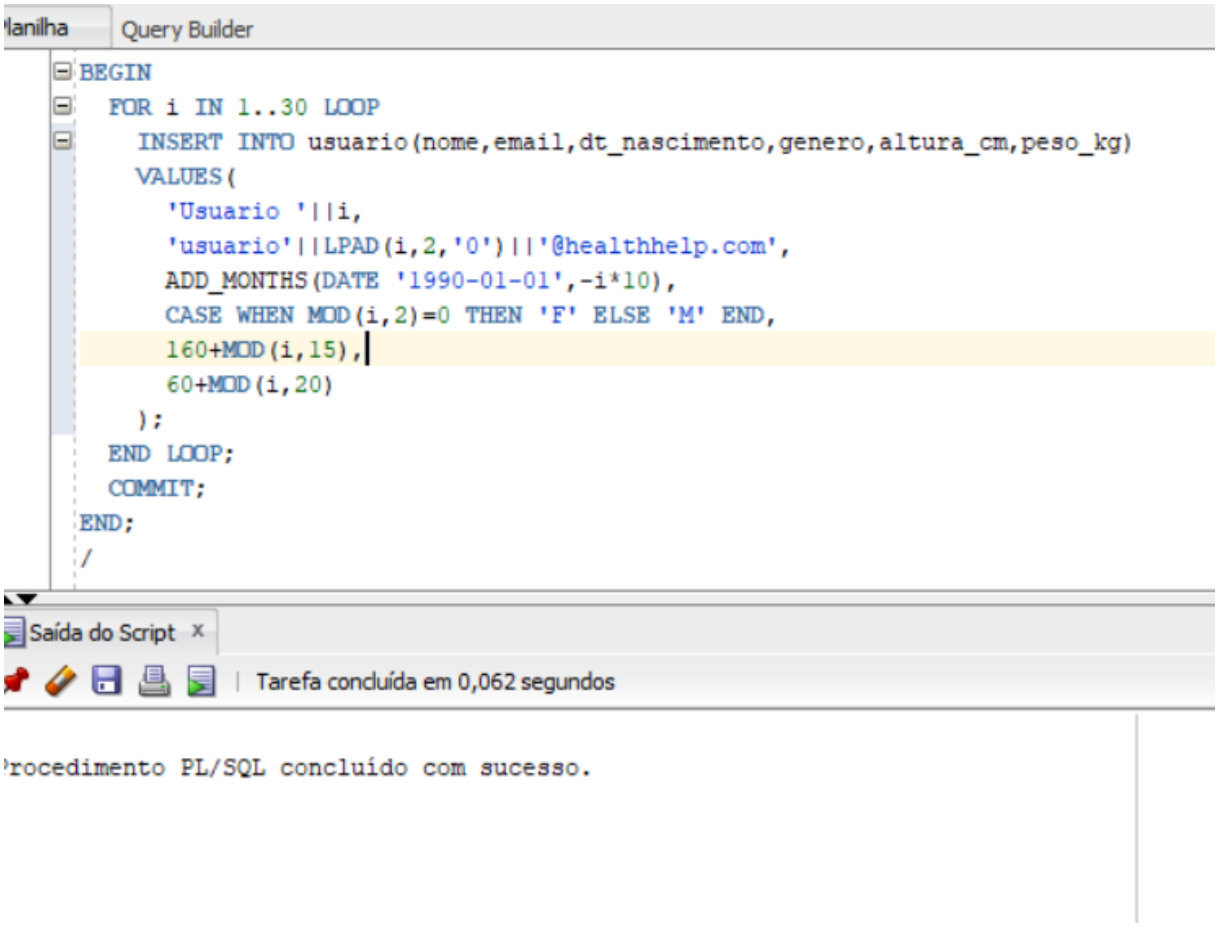
```
Procedimento PL/SQL concluído com sucesso.

Table USUARIO criado.

Table CATEGORIA_ATIVIDADE criado.

Table REGISTRO_DIARIO criado.

Table ATIVIDADE criado.

Table HABITO criado.

Table RECOMENDACAO criado.

Table AUDIT_LOG criado.
```

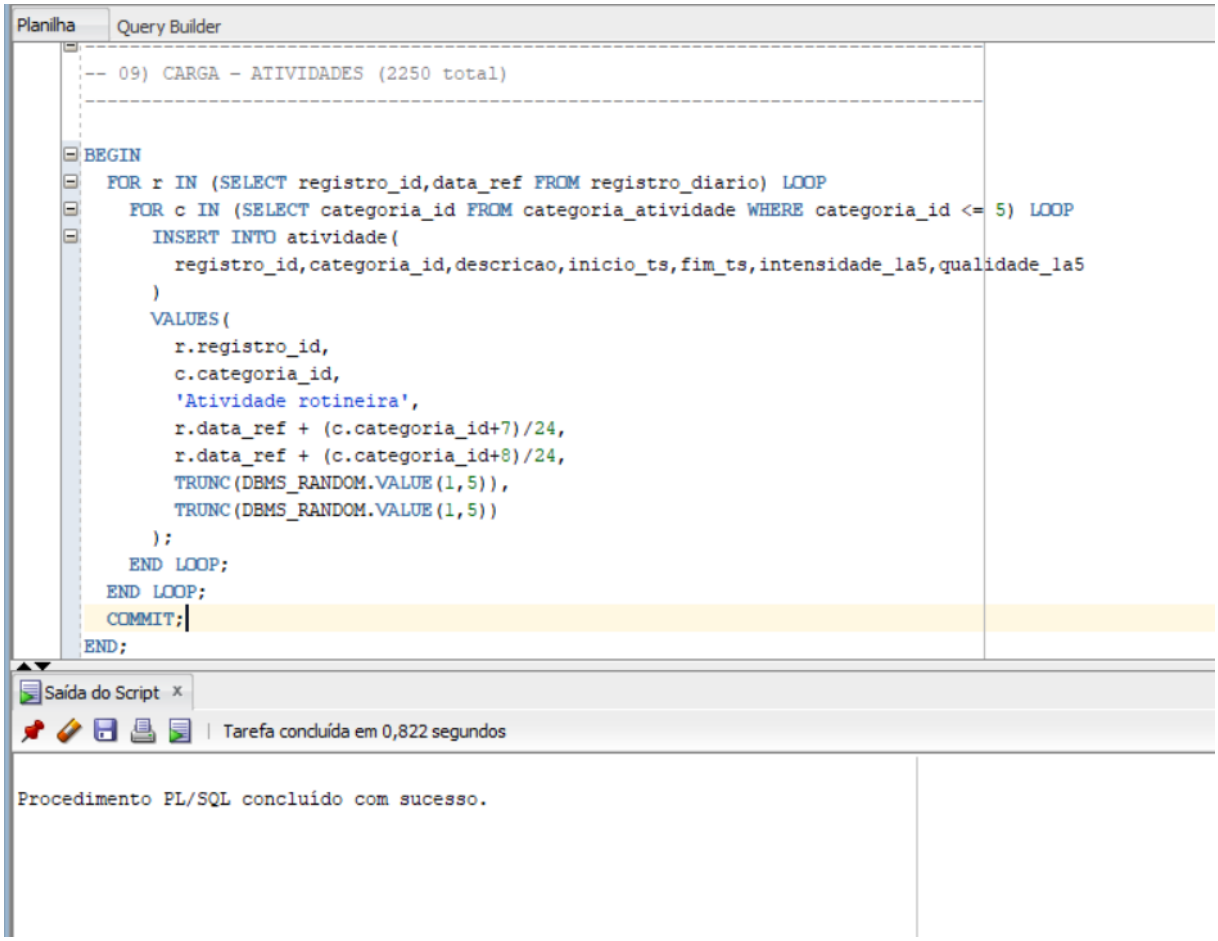
```
Trigger TRG_AUDIT_USUARIO compilado

Trigger TRG_AUDIT_REGISTRO compilado

Trigger TRG_AUDIT_ATIVIDADE compilado

Trigger TRG_AUDIT_RECOMENDACAO compilado
```

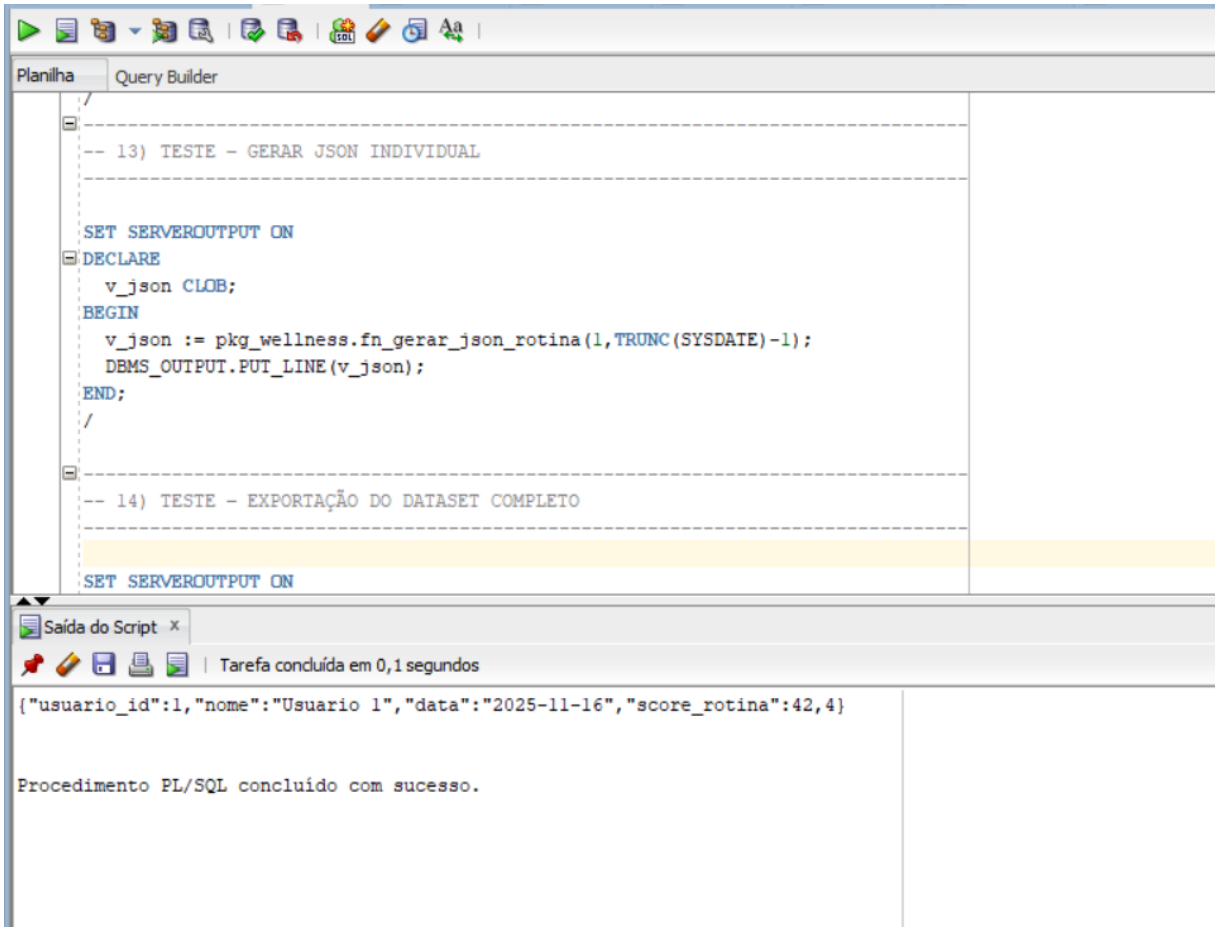
```
Package PKG_WELLNESS compilado
```

```
Package Body PKG_WELLNESS compilado
```

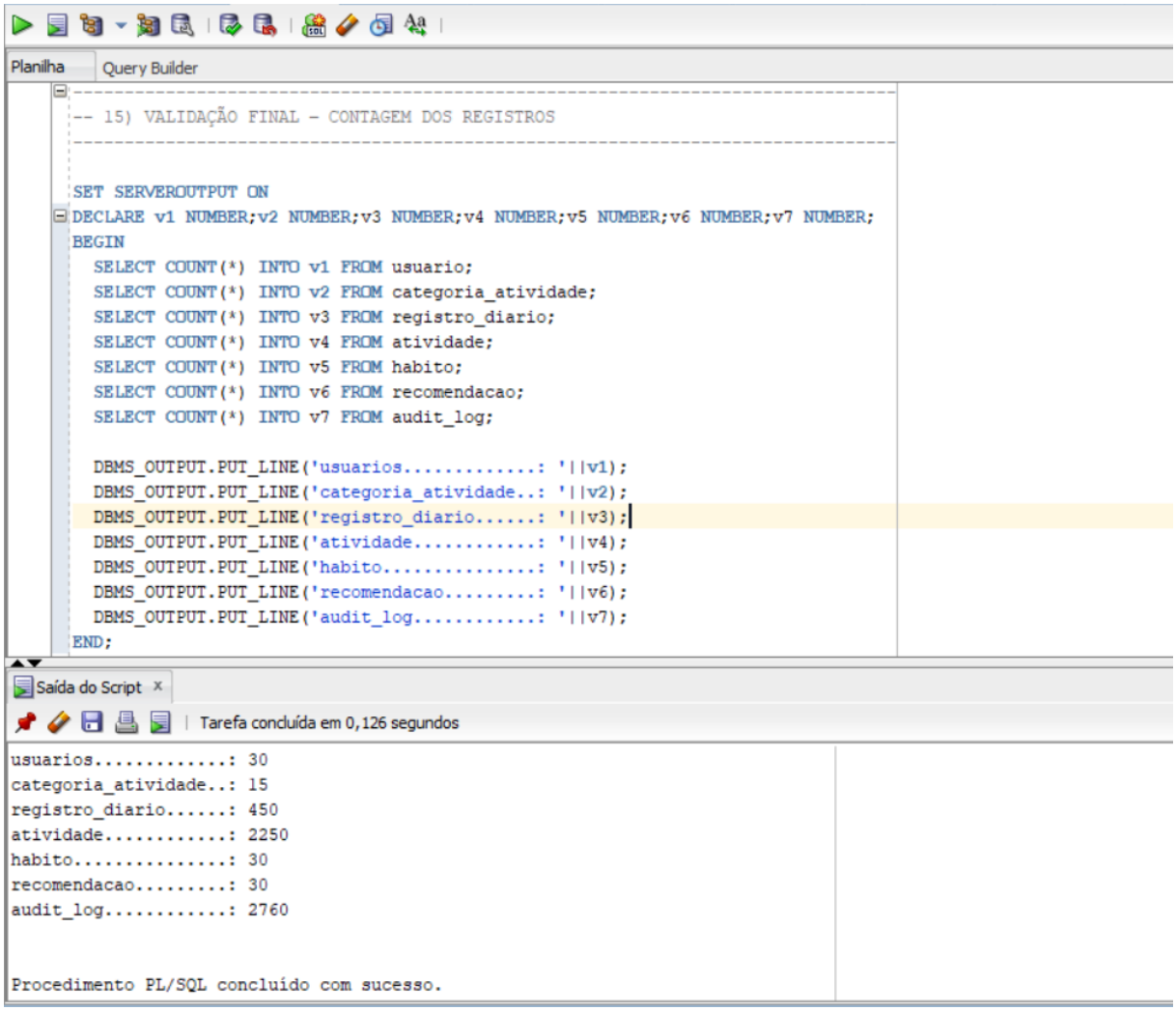
```
1 linha inserido.

1 linha inserido.

1 linha inserido.

1 linha inserido.

1 linha inserido.

1 linha inserido.

1 linha inserido.

1 linha inserido.

1 linha inserido.

1 linha inserido.

1 linha inserido.

1 linha inserido.

1 linha inserido.
```


anilha
Query Builder
BEGIN
FOR i IN 1..30 LOOP
INSERT INTO usuario(nome,email,dt_nascimento,genero,altura_cm,peso_kg)
VALUES(
'Usuario '||i,
'usuario'||LPAD(i,2,'0')||'@healthhelp.com',
ADD_MONTHS(DATE '1990-01-01',-i*10),
CASE WHEN MOD(i,2)=0 THEN 'F' ELSE 'M' END,
160+MOD(i,15),
60+MOD(i,20)
);
END LOOP;
COMMIT;
END;
/
Saída do Script
Tarefa concluída em 0,062 segundos
rocedimento PL/SQL concluído com sucesso.


Planilha
Query Builder
-- 09) CARGA – ATIVIDADES (2250 total)
BEGIN
FOR r IN (SELECT registro_id,data_ref FROM registro_diario) LOOP
FOR c IN (SELECT categoria_id FROM categoria_atividade WHERE categoria_id <= 5) LOOP
INSERT INTO atividade(
registro_id,categoria_id,descricao,inicio_ts,fim_ts,intensidade_la5,qualidade_la5
)
VALUES(
r.registro_id,
c.categoria_id,
'Atividade rotineira',
r.data_ref + (c.categoria_id+7)/24,
r.data_ref + (c.categoria_id+8)/24,
TRUNC(DBMS_RANDOM.VALUE(1,5)),
TRUNC(DBMS_RANDOM.VALUE(1,5))
);
END LOOP;
END LOOP;
COMMIT;
END;
Saída do Script
Tarefa concluída em 0,822 segundos
Procedimento PL/SQL concluído com sucesso.


Planilha
Query Builder
/
-- 13) TESTE - GERAR JSON INDIVIDUAL
SET SERVEROUTPUT ON
DECLARE
v_json CLOB;
BEGIN
v_json := pkg_wellness.fn_gerar_json_rotina(1,TRUNC(SYSDATE)-1);
DBMS_OUTPUT.PUT_LINE(v_json);
END;
/
-- 14) TESTE - EXPORTAÇÃO DO DATASET COMPLETO
SET SERVEROUTPUT ON
Saída do Script
Tarefa concluída em 0,1 segundos
{"usuario_id":1,"nome":"Usuario 1","data":"2025-11-16","score_rotina":42,4}
Procedimento PL/SQL concluído com sucesso.


-- 14) TESTE - EXPORTAÇÃO DO DATASET COMPLETO
SET SERVEROUTPUT ON
DECLARE
v_json CLOB;
BEGIN
prc_export_dataset_json(v_json);
DBMS_OUTPUT.PUT_LINE(v_json);
END;
/
-- 15) VALIDAÇÃO FINAL - CONTAGEM DOS REGISTROS
Saída do Script
Tarefa concluída em 0,139 segundos
Procedimento PL/SQL concluído com sucesso.


Planilha
Query Builder
-- 15) VALIDAÇÃO FINAL – CONTAGEM DOS REGISTROS
SET SERVEROUTPUT ON
DECLARE v1 NUMBER;v2 NUMBER;v3 NUMBER;v4 NUMBER;v5 NUMBER;v6 NUMBER;v7 NUMBER;
BEGIN
SELECT COUNT(*) INTO v1 FROM usuario;
SELECT COUNT(*) INTO v2 FROM categoria_atividade;
SELECT COUNT(*) INTO v3 FROM registro_diario;
SELECT COUNT(*) INTO v4 FROM atividade;
SELECT COUNT(*) INTO v5 FROM habito;
SELECT COUNT(*) INTO v6 FROM recomendacao;
SELECT COUNT(*) INTO v7 FROM audit_log;
DBMS_OUTPUT.PUT_LINE('usuarios.....: '||v1);
DBMS_OUTPUT.PUT_LINE('categoria_atividade..: '||v2);
DBMS_OUTPUT.PUT_LINE('registro_diario.....: '||v3);
DBMS_OUTPUT.PUT_LINE('atividade.....: '||v4);
DBMS_OUTPUT.PUT_LINE('habito.....: '||v5);
DBMS_OUTPUT.PUT_LINE('recomendacao.....: '||v6);
DBMS_OUTPUT.PUT_LINE('audit_log.....: '||v7);
END;
Saída do Script
Tarefa concluída em 0,126 segundos
usuarios.....: 30
categoria_atividade..: 15
registro_diario.....: 450
atividade.....: 2250
habito.....: 30
recomendacao.....: 30
audit_log.....: 2760
Procedimento PL/SQL concluído com sucesso.

```
>_MONGOSH
mongosh: HealthHelp_Local
    dt_nasciment
    altura_cm: 168,
    peso_kg: 68,
    media_score_rotina: 67.6
  }
  {
    _id: ObjectId('691b6d454bdd599438aa927f'),
    usuario_id: 19,
    nome: 'Usuario 19',
    email: 'usuario19@healthhelp.com',
    genero: 'M',
    dt_nascimento: '1971-08-01',
    altura_cm: 169,
    peso_kg: 69,
    media_score_rotina: 68.3
  }
  {
    _id: ObjectId('691b6d454bdd599438aa9280'),
    usuario_id: 20,
    nome: 'Usuario 20',
    email: 'usuario20@healthhelp.com',
    genero: 'F',
    dt_nascimento: '1970-09-01',
    altura_cm: 160,
    peso_kg: 60,
    media_score_rotina: 69
  }
  Type "it" for more
healthhelp_db>
```

>_MONGOSH

mongosh: HealthHelp_Local

```
}
{
  _id: ObjectId('691b6d454bdd599438aa9279'),
  usuario_id: 13,
  nome: 'Usuario 13',
  email: 'usuario13@healthhelp.com',
  genero: 'M',
  dt_nascimento: '1977-02-01',
  altura_cm: 163,
  peso_kg: 63,
  media_score_rotina: 64.1
}
{
  _id: ObjectId('691b6d454bdd599438aa927a'),
  usuario_id: 14,
  nome: 'Usuario 14',
  email: 'usuario14@healthhelp.com',
  genero: 'F',
  dt_nascimento: '1976-03-01',
  altura_cm: 164,
  peso_kg: 64,
  media_score_rotina: 64.8
}
{
  _id: ObjectId('691b6d454bdd599438aa927b'),
  usuario_id: 15,
  nome: 'Usuario 15',
  email: 'usuario15@healthhelp.com',
  genero: 'M',
```

>_MONGOSH

mongosh: HealthHelp_Local

```
    usuario_id: 15,
    nome: 'Usuario 15',
    email: 'usuario15@healthhelp.com',
    genero: 'M',
    dt_nascimento: '1975-04-01',
    altura_cm: 165,
    peso_kg: 65,
    media_score_rotina: 65.5
  }
  {
    _id: ObjectId('691b6d454bdd599438aa927c'),
    usuario_id: 16,
    nome: 'Usuario 16',
    email: 'usuario16@healthhelp.com',
    genero: 'F',
    dt_nascimento: '1974-05-01',
    altura_cm: 166,
    peso_kg: 66,
    media_score_rotina: 66.2
  }
  {
    _id: ObjectId('691b6d454bdd599438aa927d'),
    usuario_id: 17,
    nome: 'Usuario 17',
    email: 'usuario17@healthhelp.com',
    genero: 'M',
    dt_nascimento: '1973-06-01',
    altura_cm: 167,
    peso_kg: 67,
```

>_MONGOSH

```
{
  _id: ObjectId('691b6d454bdd599438aa9275'),
  usuario_id: 9,
  nome: 'Usuario 9',
  email: 'usuario09@healthhelp.com',
  genero: 'M',
  dt_nascimento: '1981-10-01',
  altura_cm: 169,
  peso_kg: 69,
  media_score_rotina: 61.3
}
{
  _id: ObjectId('691b6d454bdd599438aa9276'),
  usuario_id: 10,
  nome: 'Usuario 10',
  email: 'usuario10@healthhelp.com',
  genero: 'F',
  dt_nascimento: '1980-11-01',
  altura_cm: 160,
  peso_kg: 60,
  media_score_rotina: 62
}
{
  _id: ObjectId('691b6d454bdd599438aa9277'),
  usuario_id: 11,
  nome: 'Usuario 11',
  email: 'usuario11@healthhelp.com',
  genero: 'M',
  dt_nascimento: '1979-12-01',
```

>_MONGOSH

mongosh: HealthHelp_Local

```
}
{
  _id: ObjectId('691b6d454bdd599438aa9272'),
  usuario_id: 6,
  nome: 'Usuario 6',
  email: 'usuario06@healthhelp.com',
  genero: 'F',
  dt_nascimento: '1984-07-01',
  altura_cm: 166,
  peso_kg: 66,
  media_score_rotina: 59.2
}
{
  _id: ObjectId('691b6d454bdd599438aa9273'),
  usuario_id: 7,
  nome: 'Usuario 7',
  email: 'usuario07@healthhelp.com',
  genero: 'M',
  dt_nascimento: '1983-08-01',
  altura_cm: 167,
  peso_kg: 67,
  media_score_rotina: 59.9
}
{
  _id: ObjectId('691b6d454bdd599438aa9274'),
  usuario_id: 8,
  nome: 'Usuario 8',
  email: 'usuario08@healthhelp.com',
  genero: 'F',
```

>_MONGOSH

mongosh: HealthHelp_Local

```
}
{
  _id: ObjectId('691b6d454bdd599438aa926f'),
  usuario_id: 3,
  nome: 'Usuario 3',
  email: 'usuario03@healthhelp.com',
  genero: 'M',
  dt_nascimento: '1987-04-01',
  altura_cm: 163,
  peso_kg: 63,
  media_score_rotina: 57.1
}
{
  _id: ObjectId('691b6d454bdd599438aa9270'),
  usuario_id: 4,
  nome: 'Usuario 4',
  email: 'usuario04@healthhelp.com',
  genero: 'F',
  dt_nascimento: '1986-05-01',
  altura_cm: 164,
  peso_kg: 64,
  media_score_rotina: 57.8
}
{
  _id: ObjectId('691b6d454bdd599438aa9271'),
  usuario_id: 5,
  nome: 'Usuario 5',
  email: 'usuario05@healthhelp.com',
  genero: 'M',
```

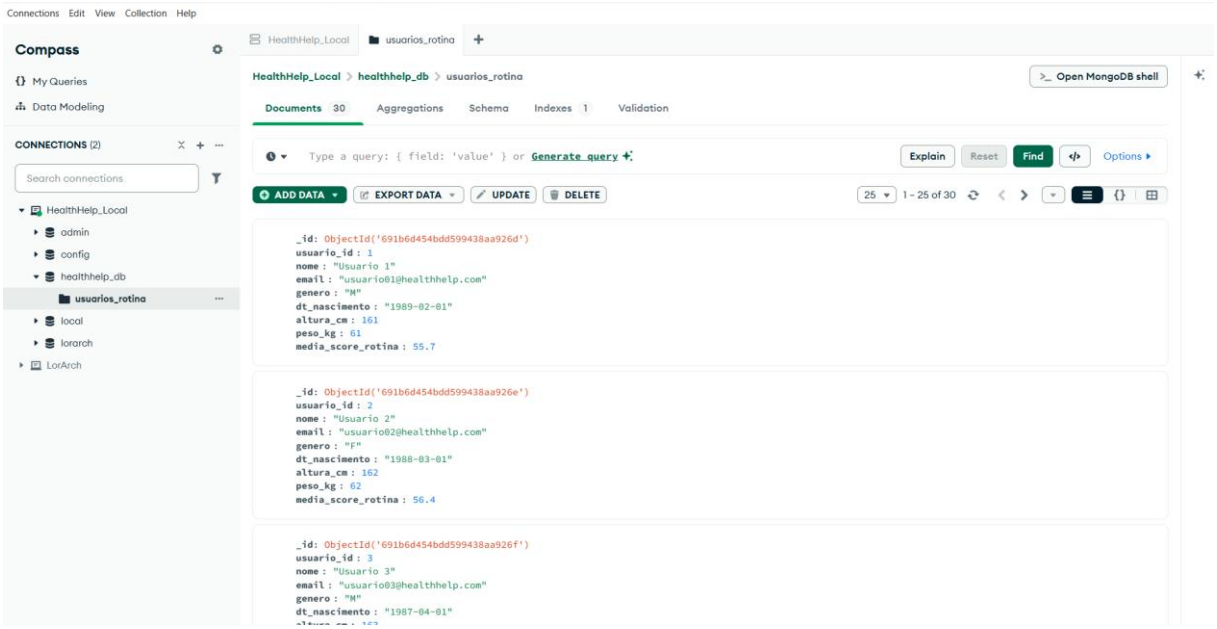
>_MONGOSH

```
> use("healthhelp_db")

  db.usuarios_rotina.createIndex(
    { email: 1 },
    { unique: true, name: "UK_usuario_email" }
  )

  db.usuarios_rotina.createIndex(
    { media_score_rotina: -1 },
    { name: "IDX_score_desc" }
  )
  db["usuarios_rotina"].find()
< {
    _id: ObjectId('691b6d454bdd599438aa926d'),
    usuario_id: 1,
    nome: 'Usuario 1',
    email: 'usuario01@healthhelp.com',
    genero: 'M',
    dt_nascimento: '1989-02-01',
    altura_cm: 161,
    peso_kg: 61,
    media_score_rotina: 55.7
  }
  {
    _id: ObjectId('691b6d454bdd599438aa926e'),
    usuario_id: 2,
    nome: 'Usuario 2',
    email: 'usuario02@healthhelp.com',
    genero: 'F',
```

>_MONGOSH

```
> use healthhelp_db
< switched to db healthhelp_db
> db.usuarios_rotina.find(
    {},
    { _id: 0, nome: 1, media_score_rotina: 1 }
  ).sort({ media_score_rotina: -1 }).limit(5)
< {
    nome: 'Usuario 30',
    media_score_rotina: 76
  }
  {
    nome: 'Usuario 29',
    media_score_rotina: 75.3
  }
  {
    nome: 'Usuario 28',
    media_score_rotina: 74.6
  }
  {
    nome: 'Usuario 27',
    media_score_rotina: 73.9
  }
  {
    nome: 'Usuario 26',
    media_score_rotina: 73.2
  }
healthhelp_db>
```

```
>_MONGOSH

> use healthhelp_db
< switched to db healthhelp_db
> db.usuarios_rotina.find(
    { media_score_rotina: { $lt: 60 } },
    { _id: 0, nome: 1, media_score_rotina: 1 }
  )
< {
    nome: 'Usuario 7',
    media_score_rotina: 59.9
  }
  {
    nome: 'Usuario 6',
    media_score_rotina: 59.2
  }
  {
    nome: 'Usuario 5',
    media_score_rotina: 58.5
  }
  {
    nome: 'Usuario 4',
    media_score_rotina: 57.8
  }
  {
    nome: 'Usuario 3',
    media_score_rotina: 57.1
  }
  {
    nome: 'Usuario 2',
    media_score_rotina: 56.4
```



HealthHelp_Local > healthhelp_db > usuarios_rotina
Open MongoDB shell
Documents 30
Aggregations
Schema
Indexes 1
Validation
Type a query: { field: 'value' } or Generate query
Explain
Reset
Find
Options
ADD DATA
EXPORT DATA
UPDATE
DELETE
25
1 - 25 of 30

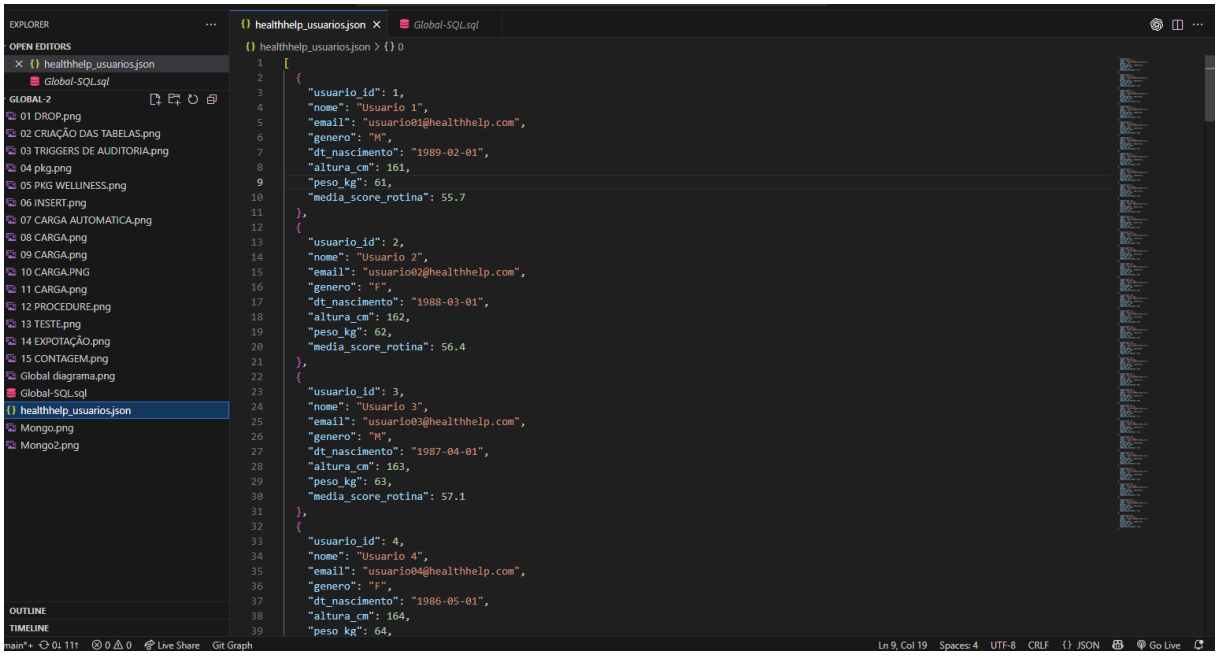

```
_id: ObjectId('691b6d454bdd599438aa926d')
usuario_id : 1
nome : "Usuario 1"
email : "usuario01@healthhelp.com"
genero : "M"
dt_nascimento : "1989-02-01"
altura_cm : 161
peso_kg : 61
media_score_rotina : 55.7
```

```
_id: ObjectId('691b6d454bdd599438aa926e')
usuario_id : 2
nome : "Usuario 2"
email : "usuario02@healthhelp.com"
genero : "F"
dt_nascimento : "1988-03-01"
altura_cm : 162
peso_kg : 62
media_score_rotina : 56.4
```

```
_id: ObjectId('691b6d454bdd599438aa926f')
usuario_id : 3
nome : "Usuario 3"
email : "usuario03@healthhelp.com"
genero : "M"
dt_nascimento : "1987-04-01"
altura_cm : 163
```

>_MONGOSH

```
> use healthhelp_db
< switched to db healthhelp_db
> db.usuarios_rotina.aggregate([
    { $group: {
        _id: "$genero",
        quantidade: { $sum: 1 },
        media_score: { $avg: "$media_score_rotina" }
    }}
  ])
< {
    _id: 'F',
    quantidade: 15,
    media_score: 66.2
  }
  {
    _id: 'M',
    quantidade: 15,
    media_score: 65.5
  }
healthhelp_db>
```


EXPLORER
OPEN EDITORS
healthhelp_usuarios.json
Global-SQL.sql
GLOBAL-2
01 DROP.png
02 CRIAÇÃO DAS TABELAS.png
03 TRIGGERS DE AUDITORIA.png
04 pkg.png
05 PKG WELLINESS.png
06 INSERT.png
07 CARGA AUTOMATICA.png
08 CARGA.png
09 CARGA.png
10 CARGA.PNG
11 CARGA.png
12 PROCEDURE.png
13 TESTE.png
14 EXPOTAÇÃO.png
15 CONTAGEM.png
Global diagrama.png
Global-SQL.sql
healthhelp_usuarios.json
Mongo.png
Mongo2.png
OUTLINE
TIMELINE
healthhelp_usuarios.json > {} 0
[
{
"usuario_id": 1,
"nome": "Usuario 1",
"email": "usuario01@healthhelp.com",
"genero": "M",
"dt_nascimento": "1989-02-01",
"altura_cm": 161,
"peso_kg": 61,
"media_score_rotina": 55.7
},
{
"usuario_id": 2,
"nome": "Usuario 2",
"email": "usuario02@healthhelp.com",
"genero": "F",
"dt_nascimento": "1988-03-01",
"altura_cm": 162,
"peso_kg": 62,
"media_score_rotina": 56.4
},
{
"usuario_id": 3,
"nome": "Usuario 3",
"email": "usuario03@healthhelp.com",
"genero": "M",
"dt_nascimento": "1987-04-01",
"altura_cm": 163,
"peso_kg": 63,
"media_score_rotina": 57.1
},
{
"usuario_id": 4,
"nome": "Usuario 4",
"email": "usuario04@healthhelp.com",
"genero": "F",
"dt_nascimento": "1986-05-01",
"altura_cm": 164,
"peso_kg": 64,
Live Share
Git Graph
Ln 9, Col 19
Spaces: 4
UTF-8
CRLF
JSON
Go Live

>_MONGOSH

```
> use healthhelp_db
< switched to db healthhelp_db
> db.usuarios_rotina.aggregate([
    {
      $bucket: {
        groupBy: "$media_score_rotina",
        boundaries: [50,60,70,80,90,100],
        default: ">=100",
        output: { qtd: { $sum: 1 } }
      }
    }
  ])
< {
    _id: 50,
    qtd: 7
  }
  {
    _id: 60,
    qtd: 14
  }
  {
    _id: 70,
    qtd: 9
  }
healthhelp_db >
```

18. Links – GitHub e YouTube

Nesta seção serão disponibilizados os links oficiais do projeto para consulta do código-fonte e da apresentação em vídeo.

GitHub do Projeto HealthHelp:	https://github.com/Ramalho044/Global-BD.git
Vídeo de Apresentação (YouTube):	https://youtu.be/A_vERL1FEng

Esses links servem como comprovação adicional da implementação, execução e apresentação do sistema, complementando a documentação escrita com demonstrações práticas.