

Neste segundo componente queremos apresentar como o nosso projeto se vai agrupar e como certos modelos e esquemas vão ser realizados pelo nosso grupo.

Vamos realizar um modelo conceitual de dados (A4), um esquema relacional, validação e refinamento desse esquema (A5) e vamos indicar indexes, triggers e os dados da base de dados.

O Modelo de Domínio Conceitual contém a identificação e descrição das entidades do domínio e os relacionamentos entre elas em um diagrama de classes UML. A figura 1 representa as principais entidades organizacionais, as relações entre elas, os atributos e a multiplicidade de relações para o nosso projeto, “O Ardina”.

[illegible]

1 / 25

2. Additional Business Rules

Regras e restrições adicionais que não podem ser transmitidas no diagrama de classes UML.

- **BR01**- Utilizador não pode fazer report de elementos seus (próprio utilizador, notícias ou comentários) ;
- **BR02**- Utilizador não pode reagir (colocar gosto ou não) nos seus elementos (notícias ou comentários) ;
- **BR03**- Utilizador não pode colocar nos favoritos algo que ele é proprietário (tag não tem proprietário) ;
- **BR04**- Utilizador não se pode seguir a ele próprio ;
- **BR05**- Só o utilizador com permissão 'a' (Administrador) pode inserir FAQ, eliminar qualquer elemento do site (utilizador, notícia, comentário e tag) ;
- **BR06**- A data do comentário tem que ser posterior a data da notícia;
- **BR07**- A data da notificação tem que ser a mesma da ação (por exemplo, a notificação do comentário tem a mesma data da inserção do comentário) ;
- **BR08**- Só o utilizador proprietário pode fazer alterações (editar ou apagar) os seus elementos (notícias, comentários e votos), exceto o Administrador que pode apagar tudo ;
- **BR09**- Só o Administrador pode banir algum utilizador ;

A5: Esquema Relacional, validação e refinação do esquema

Este artefacto contém o esquema relacional obtido pelo mapeamento do modelo de dados conceptuais descrito no A4. O esquema relacional inclui o esquema de relação, atributos, domínios, PRIMARY KEY, FOREIGN KEY e outras regras de integridade tais como: UNIQUE, DEFAULT, NOT NULL, CHECK.

1. Esquema Relacional

Esquemas relacionais estão em especificados em notação compacta:

(Como este relatório está em formato Markdown, a ferramenta para sublinhar não funciona, então para as chaves primárias colocamos em negrito e todas as regras colocamos em capslock.)

Relação de referência	Notação compacta da referência
R01	utilizador (id , nome UK NN, email UK NN, password NN, foto, permissao NN CK permissao IN UserType, contacto NN)
R02	noticia(id , autor_id -> utilizador, titulo NN, descricao NN, data NN DF Today)
R03	tag(id , nome NN UK, prioridade NN CK prioridade)

Relação de referência	Notação compacta da referência
R04	comentario(id , autor_id -> utilizador, not_id -> noticia NN, texto NN, data NN DF Today)
R05	categoria(not_id -> noticia, tag_id -> tag)
R06	faq(id , autor_id -> utilizador, questao NN, resposta NN)
R07	imagem(id , legenda NN, imag_path UK NN, not_id -> noticia)
R08	vot_com(id , com_id -> comentario NN, autor_id -> utilizador, tipo NN CK tipo IN Gosto)
R09	vot_not(id , not_id -> noticia, autor_id -> utilizador, tipo NN CK tipo IN Gosto)
R10	fav_not(autor_id -> utilizador, not_id -> noticia)
R11	fav_com(autor_id -> utilizador, comentario_id -> comentario)
R12	fav_tag(autor_id -> utilizador, tag_id -> tag)
R13	n_seguidor(followed_id -> utilizador, infos_id -> utilizador, lido NN DF FALSE, data NN DF Today)
R14	n_vot_not(autor_id -> utilizador, voto_id -> vot_not, lido NN DF FALSE, data NN DF Today)
R15	n_vot_com(autor_id -> utilizador, voto_id -> vot_com, lido NN DF FALSE, data NN DF Today)
R16	n_comentario(autor_id -> utilizador, com_id -> comentario, lido NN DF FALSE, data NN DF Today)
R17	n_uti_bloq(id , bloq_id -> utilizador, motivo NN, lido NN DF FALSE, data NN DF Today)
R18	report_n(id , autor_id -> utilizador, not_id -> noticia, tiporesp_id -> texto_report , resolvido NN DF FALSE)
R19	report_c(id , autor_id -> utilizador, comentario_id -> comentario, tiporesp_id -> texto_report, resolvido NN DF FALSE)
R20	report_u(id , autor_id -> utilizador, uti_id -> utilizador, tiporesp_id -> texto_report NN, resolvido NN DF FALSE)
R21	texto_report(id , report NN)
R22	info_seguidor(followed_id -> utilizador, infos_id -> utilizador)
R23	publicidade(id , imagem UK NN)

Legenda: UK = UNIQUE, NN = NOT NULL, CK = CHECK,DF = DEFAULT

1.1. Domínios

UserType	ENUM ('a', 'u', 'b')
Gosto	ENUM('like', 'dislike')
Today	DATE DEFAULT CURRENT_DATE

1.2. Validação de Esquema

Para validar o Esquema Relacional obtido do Modelo Conceitual, todas as dependências funcionais são identificadas e a normalização de todos os esquemas de relação é realizada.

Tabela R01 (utilizador)

Keys : {id}, {email}, {nome}	
Dependências Funcionais	
FD0101	{id} -> {nome, email, password, foto, permissao, contacto}
FD0102	{email} -> {id, nome, password, foto, permissao, contacto}
FD0103	{nome} -> { id, email, password, foto, permissao, contacto}
Forma Normal	BCNF

Tabela R02 (noticia)

Keys: {id}	
Dependências Funcionais	
FD0201	{id} -> {autor_id, titulo, descricao, data}
Forma Normal	BCNF

Tabela R03 (tag)

Keys: {id}, {nome}	
Dependências Funcionais	
FD0301	{id} -> {nome, prioridade}
FD0302	{nome} -> {id, prioridade}
Forma Normal	BCNF

Tabela R04 (comentario)

Keys: {id}	
Dependências Funcionais	
FD0401	{id} -> {autor_id, not_id, texto,data}
Forma Normal	BCNF

Tabela R05 (categoria)**Keys:** {not_id, tag_id}**Dependências Funcionais**

FD0501 none

Forma Normal BCNF**Tabela R06 (faq)****Keys:** {id}, {questao}**Dependências Funcionais**

FD0601 {id} -> {autor_id, questao, resposta}

Forma Normal BCNF**Tabela R07 (imagem)****Keys:** {id}, {imag_path}**Dependências Funcionais**

FD0701 {id} -> {legenda, imag_path, not_id}

FD0702 {imag_path} -> {id,legenda, not_id}

Forma Normal BCNF**Tabela R08 (vot_com)****Keys:** {id}**Dependências Funcionais**

FD0801 {id} -> {com_id, autor_id, tipo}

Forma Normal BCNF**Tabela R09 (vot_not)****Keys:** {id}**Dependências Funcionais**

FD0901 {id} -> {not_id, autor_id, tipo}

Forma Normal BCNF**Tabela R10 (fav_not)****Keys:** {autor_id, not_id}**Dependências Funcionais**

FD1001 none

Forma Normal BCNF

Tabela R11 (fav_com)**Keys:** {autor_id, com_id}**Dependências Funcionais**

FD1101 none

Forma Normal BCNF**Tabela R12 (fav_tag)****Keys:** {autor_id, tag_id}**Dependências Funcionais**

FD1201 none

Forma Normal BCNF**Tabela R13 (n_seguidor)****Keys:** {followed_id, infos_id}**Dependências Funcionais**

FD1301 {followed_id, infos_id} -> {lido,data}

Forma Normal BCNF**Tabela R14 (n_vot_not)****Keys:** {autor_id, voto_id}**Dependências Funcionais**

FD1401 {autor_id, voto_id} -> {lido, data}

Forma Normal BCNF**Tabela R15 (n_vot_com)****Keys:** {autor_id, voto_id}**Dependências Funcionais**

FD1501 {autor_id, voto_id} -> {lido, data}

Forma Normal BCNF**Tabela R16 (n_comentario)****Keys:** {autor_id, com_id}**Dependências Funcionais**

FD1601 {autor_id, com_id} -> {lido, data}

Forma Normal BCNF**Tabela R17 (n_uti_bloq)**

Tabela R17 (n_uti_bloq)**Keys:** {id}**Dependências Funcionais**

FD1701 {id} -> {bloq_id, motivo, lido, data}

Forma Normal BCNF**Tabela R18 (report_n)****Keys:** {id}**Dependências Funcionais**

FD1801 {id} -> {autor_id, not_id, tiporesp_id, resolvido}

Forma Normal BCNF**Tabela R19 (report_c)****Keys:** {id}**Dependências Funcionais**

FD1901 {id} -> {autor_id, comentario_id, tiporesp_id, resolvido}

Forma Normal BCNF**Tabela R20 (report_u)****Keys:** {id}**Dependências Funcionais**

FD2001 {id} -> {autor_id, uti_id, tiporesp_id, resolvido}

Forma Normal BCNF**Tabela R21 (texto_report)****Keys:** {id}**Dependências Funcionais**

FD2101 {id} -> {report}

Forma Normal BCNF**Tabela R22 (info_seguidor)****Keys:** {followed_id, infos_id}**Dependências Funcionais**

FD2201 none

Forma Normal BCNF**Tabela R23 (publicidade)**

Tabela R23 (publicidade)**Keys:** {id}, {imagem}**Dependências Funcionais**

FD2301 {id} -> {imagem}

FD2302 {imagem} -> {id}

Forma Normal

BCNF

Como todas as relações estão na forma normal de Boyce-Codd (BCNF), o esquema relacional também está no BCNF e, portanto, não há necessidade de ser definido usando a normalização.

A6: Índices, integridade e base de dados preenchida

Este artefacto contém:

- Uma previsão da carga de trabalho da base de dados;
- A identificação e caracterização dos índices;
- O modelo de dados físicos da base de dados;
- O suporte de regras de integridade com triggers;
- Transições da base de dados, necessárias para garantir a integridade dos dados na presença de acessos simultâneos;
- A definição de funções na base de dados definidas pelo utilizador;
- Código SQL necessário para a criação da base de dados, tal como para a definição de todas as restrições de integridade, índices e triggers.
- Código SQL para a povoação da base de dados.

1. Carga de trabalho da Base de Dados

É fundamental quantificarmos a utilização que a nossa aplicação irá ter para que dessa maneira sejamos capazes de construir uma base de dados adequada a essa carga.

Relação	Nome da relação	Ordem de magnitude	Crescimento estimado
R01	utilizador	10k (dezenas de milhares)	10 (dezenas) / dia
R02	noticia	100k (centenas de milhares)	10 (dezenas) / dia
R03	tag	1k (milhares)	1 (unidades) / dia
R04	comentario	1M (milhões)	1k (milhares) / dia
R05	categoria	1k (milhares)	1 (unidades) / dia

Relação	Nome da relação	Ordem de magnitude	Crescimento estimado
R06	faq	10 (dezenas)	1 / dia
R07	imagem	1M (milhões)	100 (centenas) / dia
R08	vot_com	1M (milhões)	1k (milhares) / dia
R09	vot_not	10M (dezenas de milhões)	100k (centenas de milhares) / dia
R10	fav_not	10k (dezenas de milhares)	100 (centenas) / dia
R11	fav_com	10k (dezenas de milhares)	100 (centenas) / dia
R12	fav_tag	10k (dezenas de milhares)	100 (centenas) / dia
R13	n_seguidor	1M (milhões)	100 (centenas) / dia
R14	n_vor_not	10M (dezenas de milhões)	100k (centenas de milhares) / dia
R15	n_vot_com	1M (milhões)	1k (milhares) / dia
R16	n_comentario	100k (centenas de milhares)	100 (centenas) / dia
R17	n_util_bloq	1k (milhares)	10 (dezenas) / dia
R18	report_n	1M (milhões)	100 (centenas) / dia
R19	report_c	100k (centenas de milhares)	100 (centenas) / dia
R20	report_u	10k (dezenas de milhares)	10 (dezenas) / dia
R21	texto_report	1M (milhões)	1k (milhares) / dia
R22	info_seguidor	1M (milhões)	100 (centenas) / dia
R23	publicidade	100 (centenas)	1 / dia

2 - Índices Propostos

Índices são usados para reduzir os tempos de acesso numa base de dados, melhorando assim a sua performance. Apesar das vantagens, índices consomem recursos consideráveis, sendo assim necessário usa-los com algum cuidado.

2.1 Índices de Performance

Index	ID01
Relação	utilizador
Atributo	nome
Tipo	hash
Cardinalidade	alta

Index	ID01
Clustering	Não
Justificação	Obter a informação de um utilizador é executado várias vezes por isso tem que ser rápido; não precisa de suporte para consulta de intervalo; a cardinalidade é alta porque o nome de utilizador é uma chave exclusiva e, portanto, não é um bom candidato para cluster.
SQL Code	<code>CREATE INDEX username_users_idx ON utilizador USING hash (nome);</code>
Index	ID02
Relação	noticia
Atributo	data
Tipo	hash
Cardinalidade	alta
Clustering	Yes
Justificação	Para permitir a busca das primeiras 25 notícias de uma determinada data que tenham a data inferior a um determinado valor mais rapidamente; É B-tree e agrupado para permitir consultas de alcance rápido. A cardinalidade é alta porque a data pode ter vários valores
SQL Code	<code>CREATE INDEX noticias_date_idx ON noticia USING btree(data);</code>
Index	ID03
Relação	noticia
Atributo	autor_id
Tipo	hash
Cardinalidade	média
Clustering	Yes
Justificação	O atributo autor_id é usado em várias consultas, portanto, é um bom candidato para a criação de um índice. A cardinalidade é média e, como um autor pode ter vários conteúdos, é um bom candidato para agrupamento.
SQL Code	<code>CREATE INDEX autor_noticias_idx ON noticia USING hash(autor_id);</code>
Index	ID04
Relação	comentario
Atributo	not_id
Tipo	hash
Cardinalidade	alta

Index	ID04
Clustering	Não
Justificação	Obter os comentários de uma notícia é executado várias vezes, por isso tem de ser rápido. A cardinalidade é alta porque a notícia_id é uma chave exclusiva e, portanto, não é bom candidato para cluster.
SQL Code	<code>CREATE INDEX comentarios_idx ON comentario USING hash(not_id)</code>

2.2 - Índices de pesquisa Full-Text

Index	ID05
Relação	noticia
Atributo	titulo
Tipo	GIST
Cardinalidade	alta
Clustering	Não
Justificação	Usado para melhorar o desempenho de pesquisas de texto completo ao pesquisar notícias, GIST porque as notícias são dados dinâmicos.
SQL Code	<code>ALTER TABLE noticia ADD COLUMN search TSVECTOR;</code> <code>CREATE INDEX search_noticias_idx ON noticia USING GIST (search);</code>

Index	ID06
Relação	utilizador
Atributo	nome
Tipo	GIN
Cardinalidade	alta
Clustering	Não
Justificação	Usado para melhorar o desempenho de pesquisas de texto completo ao pesquisar utilizadores, porque os dados dos utilizadores raramente são alterados.
SQL Code	<code>CREATE INDEX search_users_idx ON utilizador USING GIST (search);</code>

3.Triggers

Trigger	TRIGGER01
Descrição	Acção de administrador
Justificação	Como algumas funcionalidades como bloquear users ou atualizar faqs são apenas permitidas administradores, este trigger é relevante.

SQL Code

```

CREATE OR REPLACE FUNCTION acao_de_admin() RETURNS TRIGGER AS
$BODY$
    BEGIN
        IF NOT (SELECT permissao FROM utilizador WHERE USER_TYPE = 'u'
) THEN
            RAISE EXCEPTION 'Apenas administradores podem realizar esta
operação.';
        END IF;
        RETURN utilizador;
    END
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER acao_de_admin
    BEFORE UPDATE
    FOR EACH ROW
    EXECUTE PROCEDURE acao_de_admin();

```

Trigger	TRIGGER02
Descrição	Seguir próprio utilizador
Justificação	Deverá não ser possível para um utilizador se seguir a si próprio.

SQL Code

```

CREATE OR REPLACE FUNCTION seguir_proprio() RETURNS TRIGGER AS
$BODY$
    BEGIN
        IF info_seguidor.autor_id = info_seguidor.followed_id THEN
            RAISE EXCEPTION 'Nao se pode seguir a si próprio.';
        END IF;
        RETURN info_seguidor;
    END
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER seguir_proprio
    BEFORE INSERT ON info_seguidor
    FOR EACH ROW
    EXECUTE PROCEDURE seguir_proprio();

```

Trigger	TRIGGER03
Descrição	Notificação de novo seguidor
Justificação	Um user é notificado quando um outro utilizador segue a sua conta.
SQL Code	

```
CREATE OR REPLACE FUNCTION notificacao_seguidor(followed_id, infos_id)
RETURNS TRIGGER AS
$BODY$
BEGIN
    INSERT INTO n_seguidor
    VALUES (followed_id, infos_id);

    RETURN n_seguidor;
END
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER notificacao_seguidor
AFTER INSERT ON info_seguidor
FOR EACH ROW
EXECUTE PROCEDURE notificacao_seguidor();
```

Trigger	TRIGGER04
Descrição	Notificação de comentário
Justificação	Um user é notificado quando um outro utilizador comenta a sua noticia.
SQL Code	

```
CREATE OR REPLACE FUNCTION notificacao_comentario(u_id,c_id) RETURNS
TRIGGER AS
$BODY$
BEGIN
    INSERT INTO n_comentario
    VALUES(u_id, c_id);

    RETURN n_comentario;
END
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER notificacao_comentario
AFTER INSERT ON comentario
FOR EACH ROW
EXECUTE PROCEDURE notificacao_comentario();
```

Trigger	TRIGGER05
Descrição	Notificação de voto
Justificação	Um user é notificado quando um outro utilizador vota na sua noticia.
SQL Code	

```

CREATE OR REPLACE FUNCTION notificacao_voto(voto) RETURNS TRIGGER AS
$BODY$
BEGIN
    INSERT INTO n_vot_not
    SELECT v.autor_id, v.id
    FROM vot_not v
    WHERE v.id=voto;
    RETURN n_vot_not;
END
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER notificacao_voto
AFTER INSERT ON vot_not
FOR EACH ROW
EXECUTE PROCEDURE notificacao_voto();

```

Trigger	TRIGGER06
Descrição	Atualização do feed
Justificação	Trigger usado sempre que a tabela de notícias é atualizada. Ordena-as por ordem alfabética
SQL Code	

```

CREATE OR REPLACE FUNCTION atualizar_feed() RETURNS TRIGGER AS
$BODY$
BEGIN
    SELECT *
    FROM noticia
    ORDER BY data DESC;
END
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER atualizar_feed
AFTER UPDATE ON noticia
FOR EACH ROW
EXECUTE PROCEDURE atualizar_feed();

```

4. Transactions

Transações são usadas de forma a garantir a integridade dos dados quando estes são inseridos em simultâneo.

T01	Criação de Notícia
Justificação	Para manter a consistência do conteúdo, usamos uma transação para garantir que não haja vários conteúdos sendo inseridos ao mesmo tempo. Se ocorrer um erro, um ROLLBACK é emitido.
Nível de Isolamento	Leitura repetível.
Código SQL Completo	<pre> BEGIN TRANSACTION; SET TRANSACTION ISOLATION LEVEL REPEATABLE READ; INSERT INTO noticia (autor_id, titulo, descricao, data) VALUES (\$autor_id, \$titulo, \$descricao, \$data); COMMIT; </pre>
T02	Inserir comentário
Justificação	Para manter a consistência do conteúdo, usamos uma transação para garantir que não haja vários conteúdos sendo inseridos ao mesmo tempo. Se ocorrer um erro, um ROLLBACK é emitido.
Nível de Isolamento	Leitura repetível.
Código SQL Completo	<pre> SET TRANSACTION ISOLATION LEVEL REPEATABLE READ; INSERT INTO noticia (autor_id, titulo, descricao, data) VALUES(\$autor_id, \$titulo, \$descricao, \$data) INSERT INTO comentario (autor_id, not_id, comentario) VALUES(\$autor_id, \$not_id, \$comentario) COMMIT; </pre>
T03	Reportar User

T03	Reportar User
Justificação	Para manter a consistência do conteúdo, usamos uma transação para garantir que não haja vários conteúdos sendo inseridos ao mesmo tempo. Se ocorrer um erro, um ROLLBACK é emitido.
Nível de Isolamento	Leitura repetível.
Código SQL Completo	<pre> SET TRANSACTION ISOLATION LEVEL REPEATABLE READ: INSERT INTO reportU (autor_id, users_id, tiporesp_id, resolvido) VALUES(\$autor_id, \$users_id, \$tiporesp_id, \$resolvido) COMMIT; </pre>
T04	Reportar Noticia
Justificação	Para manter a consistência do conteúdo, usamos uma transação para garantir que não haja vários conteúdos sendo inseridos ao mesmo tempo. Se ocorrer um erro, um ROLLBACK é emitido.
Nível de Isolamento	Leitura repetível.
Código SQL Completo	<pre> BEGIN TRANSACTION SET TRANSACTION ISOLATION LEVEL REPEATABLE READ: INSERT INTO report_n (autor_id, not_id, tiporesp_id) VALUES(\$autor_id, \$not_id, \$descricao, \$data); COMMIT; </pre>
T05	Reportar Comentário
Justificação	Para manter a consistência do conteúdo, usamos uma transação para garantir que não haja vários conteúdos sendo inseridos ao mesmo tempo. Se ocorrer um erro, um ROLLBACK é emitido.
Nível de Isolamento	Leitura repetível.

T05**Reportar Comentário**

Código SQL
Completo

```
BEGIN TRANSACTION

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ:

INSERT INTO reportC (autor_id, comentario_id, tiporesp_id)

VALUES($autor_id, $comentario_id, $descricao, $data);

COMMIT;
```

Annex A. Código SQL

A.1 Criação da Base de Dados

```
CREATE SCHEMA lbaw2163; SET search_path TO "lbaw2163";
```



Criação de Tipos

```
CREATE TYPE USER_TYPE AS ENUM('a','u','b');
CREATE TYPE GOSTO AS ENUM('like', 'dislike');
```



Tabelas

```
CREATE TABLE utilizador(
  id SERIAL PRIMARY KEY,
  nome VARCHAR(20) NOT NULL UNIQUE,
  email TEXT NOT NULL UNIQUE,
  password TEXT NOT NULL,
  foto TEXT,
  permissao USER_TYPE NOT NULL,
  contacto INTEGER NOT NULL,
  CONSTRAINT contacto_limites CHECK (contacto > 99999999 AND contacto <
1000000000)
);
CREATE TABLE noticia (
  id SERIAL PRIMARY KEY,
  autor_id INTEGER NOT NULL REFERENCES utilizador(id) ON DELETE CASCADE,
```

```
        titulo VARCHAR(90) NOT NULL,
        descricao TEXT NOT NULL,
        data TIMESTAMP WITH TIME ZONE DEFAULT now() NOT NULL
    );
CREATE TABLE tag (
    id SERIAL PRIMARY KEY,
    nome VARCHAR(40) NOT NULL UNIQUE,
    prioridade INTEGER NOT NULL DEFAULT 0
);
CREATE TABLE comentario (
    id SERIAL PRIMARY KEY,
    autor_id INTEGER NOT NULL REFERENCES utilizador(id) ON DELETE CASCADE,
    not_id INTEGER NOT NULL REFERENCES noticia(id) ON DELETE CASCADE,
    texto TEXT NOT NULL,
    data TIMESTAMP WITH TIME ZONE DEFAULT now() NOT NULL
);
CREATE TABLE categoria (
    not_id INTEGER NOT NULL REFERENCES noticia(id) ON DELETE CASCADE,
    tag_id INTEGER NOT NULL REFERENCES tag(id) ON DELETE CASCADE,
    PRIMARY KEY (not_id, tag_id)
);
CREATE TABLE faq (
    id SERIAL PRIMARY KEY,
    autor_id INTEGER NOT NULL REFERENCES utilizador(id) ON DELETE CASCADE,
    questao TEXT NOT NULL,
    resposta TEXT NOT NULL
);
CREATE TABLE imagem (
    id SERIAL PRIMARY KEY,
    legenda VARCHAR(30) NOT NULL,
    imag_path TEXT NOT NULL UNIQUE,
    not_id INTEGER NOT NULL REFERENCES noticia(id) ON DELETE CASCADE
);

CREATE TABLE vot_com (
    id SERIAL PRIMARY KEY,
    com_id INTEGER NOT NULL REFERENCES comentario(id) ON DELETE CASCADE,
    autor_id INTEGER NOT NULL REFERENCES utilizador(id) ON DELETE CASCADE,
    tipo GOSTO NOT NULL
);

CREATE TABLE vot_not (
    id SERIAL PRIMARY KEY,
    not_id INTEGER NOT NULL REFERENCES noticia(id) ON DELETE CASCADE,
    autor_id INTEGER NOT NULL REFERENCES utilizador(id) ON DELETE CASCADE,
    tipo GOSTO NOT NULL
);

CREATE TABLE fav_not (
    autor_id INTEGER NOT NULL REFERENCES utilizador(id) ON DELETE CASCADE,
    not_id INTEGER NOT NULL REFERENCES noticia(id) ON DELETE CASCADE,
    PRIMARY KEY(autor_id, not_id)
);
```

```
CREATE TABLE fav_com (
    autor_id INTEGER NOT NULL REFERENCES utilizador(id) ON DELETE CASCADE,
    comentario_id INTEGER NOT NULL REFERENCES comentario(id) ON DELETE
CASCADE,
    PRIMARY KEY(autor_id, comentario_id)
);

CREATE TABLE fav_tag (
    autor_id INTEGER NOT NULL REFERENCES utilizador(id) ON DELETE CASCADE,
    tag_id INTEGER NOT NULL REFERENCES tag(id) ON DELETE CASCADE,
    PRIMARY KEY(autor_id, tag_id)
);

CREATE TABLE info_seguidor (
    followed_id INTEGER NOT NULL REFERENCES utilizador(id) ON DELETE
CASCADE,
    infos_id INTEGER NOT NULL REFERENCES utilizador(id) ON DELETE CASCADE,
    PRIMARY KEY(followed_id, infos_id)
);

CREATE TABLE n_seguidor (
    followed_id INTEGER NOT NULL REFERENCES utilizador(id) ON DELETE
CASCADE,
    infos_id INTEGER NOT NULL REFERENCES utilizador(id) ON DELETE
CASCADE,
    lido BOOLEAN NOT NULL DEFAULT FALSE,
    data TIMESTAMP WITH TIME ZONE DEFAULT now() NOT NULL,
    PRIMARY KEY(followed_id, infos_id)
);

CREATE TABLE n_comentario (
    autor_id INTEGER NOT NULL REFERENCES utilizador(id) ON DELETE CASCADE,
    com_id INTEGER NOT NULL REFERENCES comentario(id) ON DELETE CASCADE,
    lido BOOLEAN NOT NULL DEFAULT FALSE,
    data TIMESTAMP WITH TIME ZONE DEFAULT now() NOT NULL,
    PRIMARY KEY(autor_id, com_id)
);

CREATE TABLE n_vot_not (
    autor_id INTEGER NOT NULL REFERENCES utilizador(id) ON DELETE CASCADE,
    voto_id INTEGER NOT NULL REFERENCES vot_not(id) ON DELETE CASCADE,
    lido BOOLEAN NOT NULL DEFAULT FALSE,
    data TIMESTAMP WITH TIME ZONE DEFAULT now() NOT NULL,
    PRIMARY KEY(autor_id, voto_id)
);

CREATE TABLE n_vot_com (
    autor_id INTEGER NOT NULL REFERENCES utilizador(id) ON DELETE CASCADE,
    voto_id INTEGER NOT NULL REFERENCES vot_com(id) ON DELETE CASCADE,
    lido BOOLEAN NOT NULL DEFAULT FALSE,
    data TIMESTAMP WITH TIME ZONE DEFAULT now() NOT NULL,
    PRIMARY KEY(autor_id, voto_id)
);
```

```
CREATE TABLE n_uti_bloq (
    id SERIAL PRIMARY KEY,
    bloq_id INTEGER NOT NULL REFERENCES utilizador(id) ON DELETE CASCADE,
    motivo TEXT NOT NULL,
    lido BOOLEAN NOT NULL DEFAULT FALSE,
    data TIMESTAMP WITH TIME ZONE DEFAULT now() NOT NULL
);

CREATE TABLE texto_report (
    id SERIAL PRIMARY KEY,
    report TEXT NOT NULL
);

CREATE TABLE report_u (
    id SERIAL PRIMARY KEY,
    autor_id INTEGER NOT NULL REFERENCES utilizador(id) ON DELETE CASCADE,
    uti_id INTEGER NOT NULL REFERENCES utilizador(id) ON DELETE CASCADE,
    tiporesp_id INTEGER NOT NULL REFERENCES texto_report(id) ON DELETE
CASCADE,
    resolvido BOOLEAN NOT NULL DEFAULT FALSE
);

CREATE TABLE report_n (
    id SERIAL PRIMARY KEY,
    autor_id INTEGER NOT NULL REFERENCES utilizador(id) ON DELETE CASCADE,
    not_id INTEGER NOT NULL REFERENCES noticia(id) ON DELETE CASCADE,
    tiporesp_id INTEGER NOT NULL REFERENCES texto_report(id) ON DELETE
CASCADE,
    resolvido BOOLEAN NOT NULL DEFAULT FALSE
);

CREATE TABLE report_c (
    id SERIAL PRIMARY KEY,
    autor_id INTEGER NOT NULL REFERENCES utilizador(id) ON DELETE CASCADE,
    comentario_id INTEGER NOT NULL REFERENCES comentario(id) ON DELETE
CASCADE,
    tiporesp_id INTEGER NOT NULL REFERENCES texto_report(id) ON DELETE
CASCADE,
    resolvido BOOLEAN NOT NULL DEFAULT FALSE
);

CREATE TABLE publicidade (
    id SERIAL PRIMARY KEY,
    imagem TEXT NOT NULL UNIQUE
);
```

A.2. População da Base de dados



Inserção nas tabelas

```
INSERT INTO utilizador (nome,email,password,foto,contacto)
VALUES
  ('Ricardo
Gomes','rgomes@hotmail.com','legionario','/src/photos/test.jpg',935978945)
,
  ('Nuno
Gaspar','gaspa2000.nuno@icloud.edu','password123','/src/photos/test1.jpg',
963390475),
  ('Sara
Cardoso','sc5298@aol.org','ricardo','/src/photos/test2.jpg',915684578),
  ('Francisco
Ribero','ribs.chics@outlook.net','19990502','/src/photos/test3.jpg',925567
441),
  ('Carolina
Antunes','carolina.antunes.official@protonmail.org','superstar','/src/phot
os/test4.jpg',911156632);
```

```
INSERT INTO noticia(autor_id,titulo,descricao)
VALUES
  (3,'Mulher perde cão em Lisboa','Luía Ferreira, separa-se de Bolinhas em
pleno Rossio, na manhã desta quarta feira. A policia já tomou conta do
caso.'),
  (3,'Joaquim Messias vai treinar o Oliveirense','O treinador Alentejano
assina por dois anos com o atual décimo classificado'),
  (2,'Os preços dos combustíveis sobem mais uma vez','É já o quinto
aumento desde o início do ano. Espera-se uma redução da gasolina no
próximo trimestre. '),
  (4,'Faculdade de Ciências considerada a melhor de Portugal','Pelo quarto
ano consecutivo, a Faculdade de Ciências é eleita a facudade de maior
sucesso no país.'),
  (2,'Ataques Russos em Miami','João Bidão condena a destruição massiva e
promete retaliação muito brevemente.');
```

```
INSERT INTO tag(nome)
VALUES
  ('Desporto'),
  ('Nacional'),
  ('Local'),
  ('Mundo');
```

```
INSERT INTO comentario(autor_id,not_id,texto)
VALUES
  (3,3,'Isto está uma vergonha. Mas eu como meto sempre 20€ não me faz
muita diferença.'),
  (3,4,'Cá para mim isso foi minado...'),
  (1,4,'Parabéns aos nossos cientistas portugueses!!!'),
  (1,1,'Espero que o encontrem depressa.'),
  (4,5,'Ouvi dizer que els estão a preparar 4 bombas atómicas.');
```

```
INSERT INTO n_comentario(autor_id,com_id)
VALUES
    (3,1),
    (3,2),
    (1,3),
    (1,4),
    (4,5);
```

```
INSERT INTO categoria(not_id, tag_id)
VALUES
    (1,3),
    (2,1),
    (3,3),
    (4,3),
    (5,4);
```

```
INSERT INTO faq(autor_id, questao, resposta)
VALUES
    (2, 'Como inserir uma notícia ?', 'Na pagina inicial clique no botão +
e de seguida escreva o titulo e a noticia nas caixas que vao aparecer. De
seguida selecione a categoria da noticia e clique publicar.');
```

```
INSERT INTO imagem(legenda,imag_path,not_id)
VALUES
    ('Bolinhas com a dona','/src/photos/test5.jpg',1),
    ('Joaquim Messias com o Presidente do
Oliveirense','/src/photos/test6.jpg',2),
    ('Fila de carros na bomba para abastecer antes da subida de
preços','/src/photos/test7.jpg',3),
    ('Departamento de ciências dos computadores da Faculdade de Ciências
da UP','/src/photos/test8.jpg',4),
    ('Exibição de equipamento bélico
americano','/src/photos/test9.jpg',5);
```

```
INSERT INTO vot_com(com_id,autor_id,tipo)
VALUES
    (1,1,'dislike'),
    (1,2,'dislike'),
    (3,2,'like'),
    (3,4,'like'),
    (2,1,'dislike'),
    (5,3,'like');
```

```
INSERT INTO n_vot_com(autor_id,voto_id)
VALUES
    (3,1),
    (3,2),
    (1,3),
    (1,4),
    (3,5),
    (4,6);
```

```
INSERT INTO vot_not(not_id,autor_id,tipo)
```

```
VALUES
    (3,3,'dislike'),
    (4,3,'dislike'),
    (2,4,'like'),
    (4,5,'like'),
    (2,1,'dislike'),
    (5,3,'like');

INSERT INTO n_vot_not(autor_id,voto_id)
VALUES
    (2,1),
    (4,2),
    (3,3),
    (4,4),
    (3,5),
    (2,6);

INSERT INTO fav_not(autor_id,not_id)
VALUES
    (5,4),
    (3,4);

INSERT INTO fav_com(autor_id,comentario_id)
VALUES
    (3,4),
    (5,3);

INSERT INTO fav_tag(autor_id,tag_id)
VALUES
    (1,1),
    (3,4);

INSERT INTO info_seguidor(followed_id, infos_id)
VALUES
    (1,3),
    (3,2),
    (4,1),
    (3,1);

INSERT INTO n_seguidor(followed_id,infos_id)
VALUES
    (1,3),
    (3,2),
    (4,1),
    (3,1);

INSERT INTO texto_report(report)
VALUES
    ('0 utilizador foi agressivo com outro e além de ser agressivo tentou
ameaçar diversos utilizadores');

INSERT INTO report_u(autor_id, uti_id, tiporesp_id)
VALUES
    (3,2,1);
```

```
INSERT INTO report_u(autor_id, uti_id, tiporesp_id, resolvido)
VALUES
    (3,2,1,TRUE);

INSERT INTO n_uti_bloq (bloq_id, motivo)
VALUES (2,'Agressividade nos comentarios.');
```

```
INSERT INTO report_n(autor_id, not_id, tiporesp_id)
VALUES
    (3,2,1);

INSERT INTO report_c(autor_id, comentario_id, tiporesp_id)
VALUES
    (3,2,1);

INSERT INTO publicidade(imagem)
VALUES
    ('/src/photos/test10.jpg'),
    ('/src/photos/test11.jpg'),
    ('/src/photos/test12.jpg');
```



Updates de valores nas tabelas

```
/*Criar um administrador*/
UPDATE utilizador
SET permissao = 'a'
WHERE id = 2;

/*Criar um user banido*/
UPDATE utilizador
SET permissao = 'b'
WHERE id = 5;

UPDATE noticia
SET titulo = 'Porto é campeão nacional!'
WHERE id = 3;

UPDATE tag
SET prioridade=1
WHERE nome = 'Nacional';
```



Eliminar valores nas tabelas


```
DELETE FROM tag  
WHERE id = 2;
```

Histórico de Revisão

Em relação à primeira submissão do trabalho, fizemos as seguintes alterações:

- Modificamos o diagrama UML, acrescentando generalizações que anteriormente estavam incorretas;
- Corrigimos a classe que tínhamos no plural;
- Correção nas relações referente ao tipo Today;
- Retirar domínios que não foram utilizados;
- Alterar o nome das relação para o formato correto (lowercase e snake_case);
- Adicionamos os script ao git;
- Corrigimos as relações na tabela de carga de trabalho na BD;
- Adicionamos a cardinalidade nos índices de performance;
- Corregimos de trigger (correção do código SQL);
- Melhoramento no código SQL (Annex A.)
 - Criação de Schema
 - Criação de tipos
 - Criação das tabelas
 - Inserção de dados nas tabelas;

Grupo 63, Data : 03/01/2022

- António Ferreira Cabral de Barbosa Campelo, up201704987@fc.up.pt
- Edgar Miguel Pinto Lourenço, up201604910@fc.up.pt
- Manuel da Silva Sá, up201805273@fc.up.pt
- Patrícia Daniela Tavares Vieira, up201805238@fc.up.pt