deti

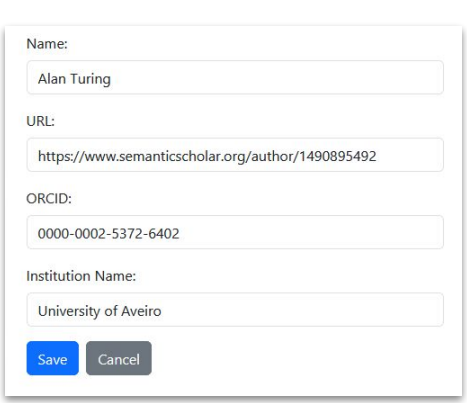
BD – Apresentação Final

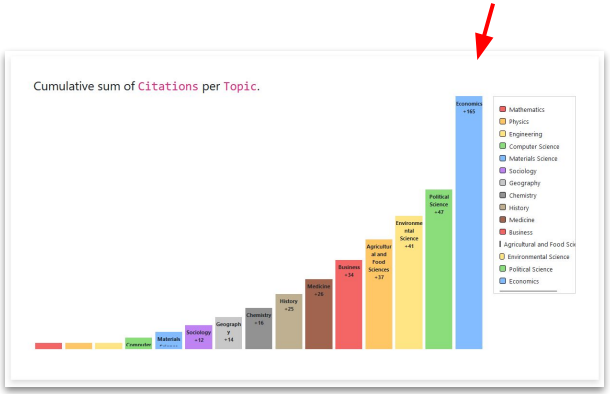
AcademiQuest

Grupo p5g1:
Pedro Pinto - 115304
João Pinto - 104384

Introdução

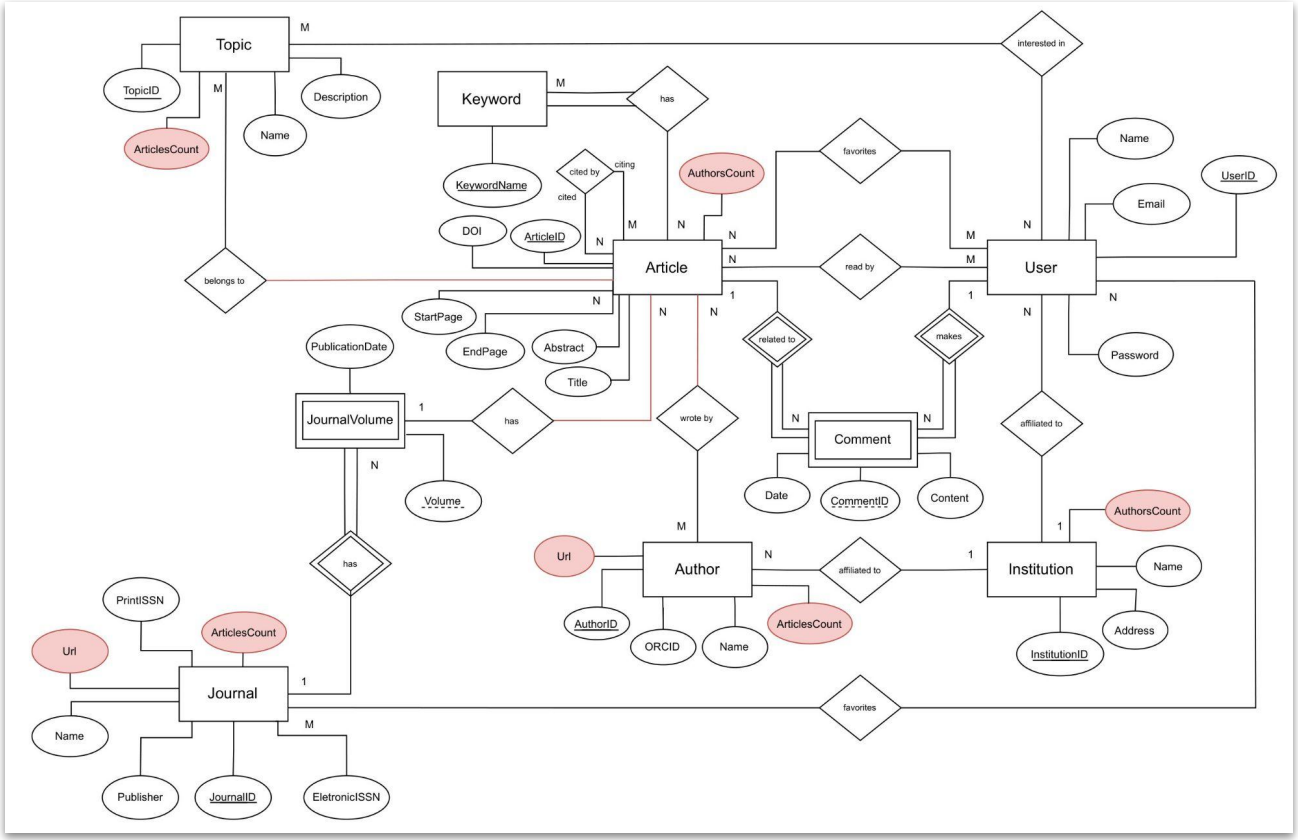
- Desenvolvimento de um sistema de **gestão de artigos científicos**.
- Entidades: **autores**, **instituições**, **artigos**, **tópicos**, **jornais**, **volume**, ...
- Utilização de **dados reais**: API Semantic Scholar.
- Interação por meio de **formulários** e apresentação de **estatísticas**.





Diag. Entidade Relacionamento

DER

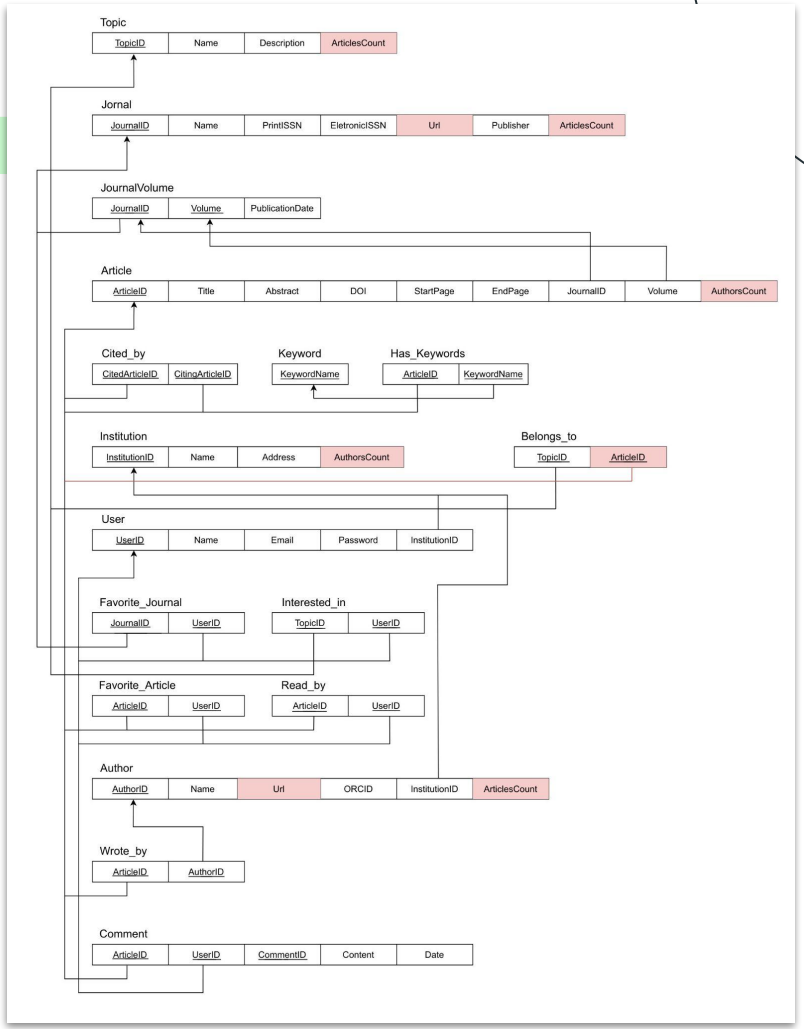


Esquema Relacional

ER

- Evitar redundância e duplicação de dados.
- Atributos *PublicationDate* e *TopicName*.
- *FK -> (JournalID, Volume)*.
- *Belongs_to* , *Cited_by* e *Wrote_by*.
- Journal contém informação comum a vários volumes.

Normalização: BCNF

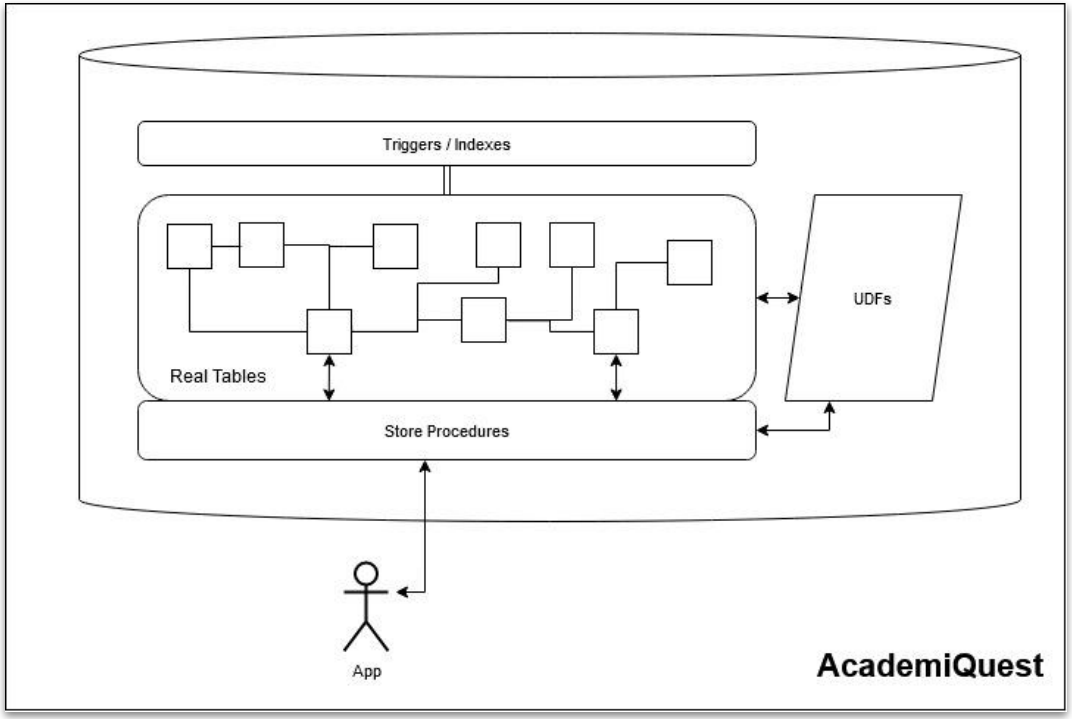


Elementos Importantes

Arquitetura

Elementos utilizados:

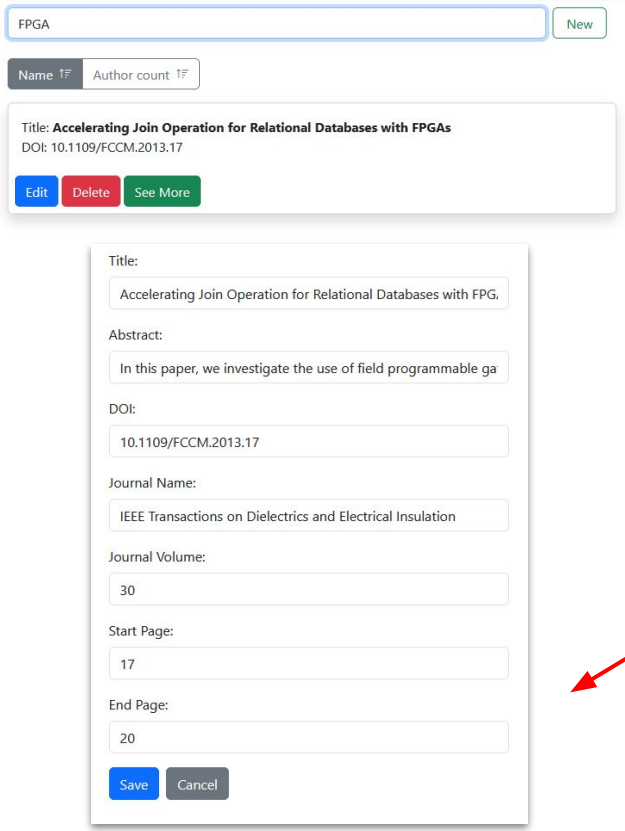
- SPs - 45
- UDFs - 6
- Indexes - 10
- Triggers - 15
- Cursors - 1



Store Procedures

E.g: Article

```
CREATE PROCEDURE DeleteArticle
    @ArticleID VARCHAR(10)
AS
BEGIN
    SET NOCOUNT ON

    BEGIN TRANSACTION
    BEGIN TRY
        DELETE FROM Wrote_by
        WHERE ArticleID = @ArticleID

        DELETE FROM Belongs_to
        WHERE ArticleID = @ArticleID

        (...)

        DELETE FROM Read_by
        WHERE ArticleID = @ArticleID

        DELETE FROM Article
        WHERE ArticleID = @ArticleID

        COMMIT TRANSACTION
    END TRY
    BEGIN CATCH
    (...)
    END CATCH
END;
```



Focadas em:

- Criar
- Atualizar
- Eliminar
- Ver mais
- Filtrar

```
CREATE PROCEDURE UpdateArticle
    (...)
AS
BEGIN
    SET NOCOUNT ON
--
    -- normalize the args
    SET @Title = NULLIF(@Title, '')
    (...)

    DECLARE @JourID VARCHAR(40)
    EXEC ValidateArticle (...), @JourID OUTPUT -- validation

    UPDATE Article
    SET (...)
    WHERE ArticleID = @ArticleID
END;
```

Triggers

- Garantir a consistência na base de dados sempre que ocorre uma modificação.
- De modo semelhante ao exemplo abaixo, foram criados triggers para as restantes entidades.

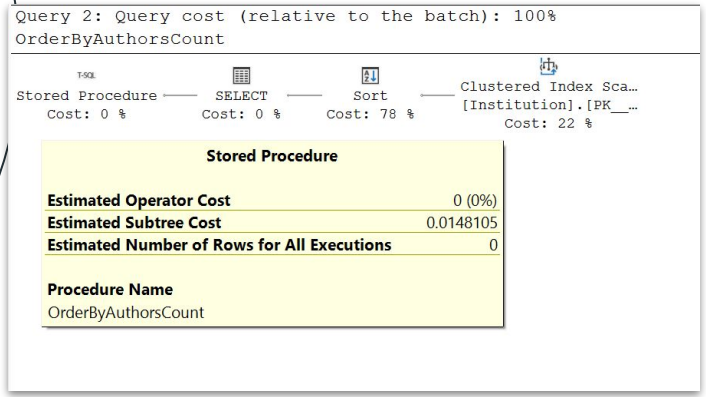
```
CREATE TRIGGER trg_author_ArticlesCount_Update
ON Wrote_by AFTER UPDATE AS
BEGIN
    -- Update previous ArticlesCount
    UPDATE Author
    SET ArticlesCount = (SELECT COALESCE(COUNT(*), 0) FROM Wrote_by WHERE Wrote_by.AuthorID = Author.AuthorID)
    FROM Author
    INNER JOIN deleted d ON Author.AuthorID = d.AuthorID

    -- Update current ArticlesCount
    UPDATE Author
    SET ArticlesCount = (SELECT COALESCE(COUNT(*), 0) FROM Wrote_by WHERE Wrote_by.AuthorID = Author.AuthorID)
    FROM Author
    INNER JOIN inserted i ON Author.AuthorID = i.AuthorID
END;
```

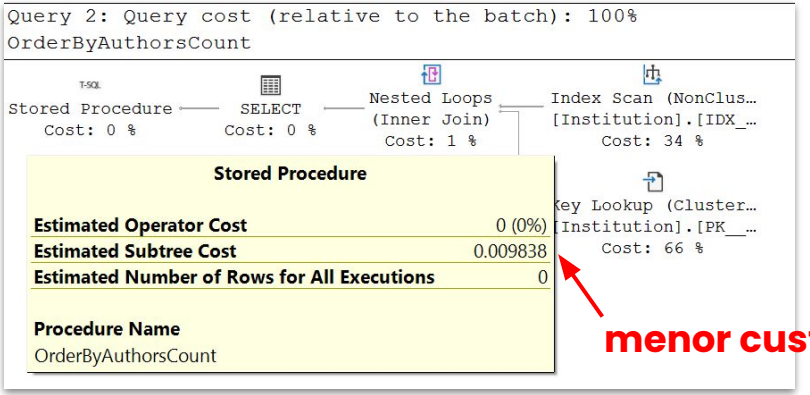
Indexes

- Melhorar o desempenho das consultas.
- Nome, número de autores e número de artigos.

```
-- Indices da tabela Institution
CREATE NONCLUSTERED INDEX IDX_Institution_Name
ON Institution ([Name]);

CREATE NONCLUSTERED INDEX IDX_Institution_AuthorsCount
ON Institution (AuthorsCount);
```

E.g: Ordenar instituições por número de autores



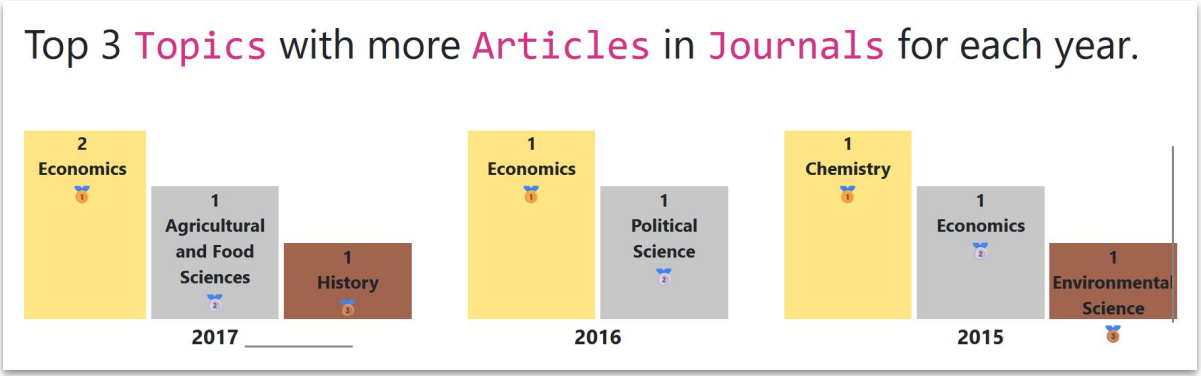


Estatísticas

UDF!!

```
CREATE PROCEDURE MostProductiveAuthorsByTopic
AS
BEGIN
    SELECT TOP 8 TopicName, AuthorName, ArticlesCount
    FROM (
        SELECT COUNT(Author.ArticlesCount) AS ArticlesCount, ROW_NUMBER() OVER (PARTITION BY TopicName ORDER BY
COUNT(Author.ArticlesCount) DESC, L.TopicName) AS AuthorRank, Author.AuthorID, Author.[Name] AS AuthorName, L.TopicName
        FROM ListAllArticlesPerTopic() AS L
        INNER JOIN Wrote_by ON Wrote_by.ArticleID = L.ArticleID
        INNER JOIN Author ON Author.AuthorID = Wrote_by.AuthorID
        GROUP BY Author.AuthorID, Author.[Name], L.TopicName
    ) AS T
    WHERE T.AuthorRank = 1
    ORDER BY T.ArticlesCount DESC
END;
```

Estatísticas

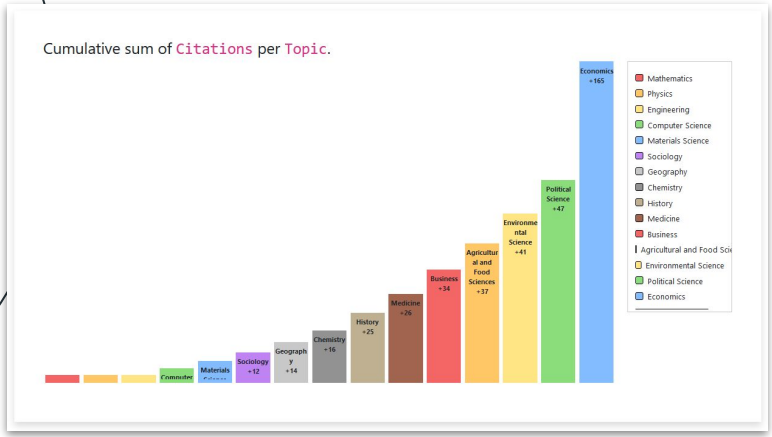


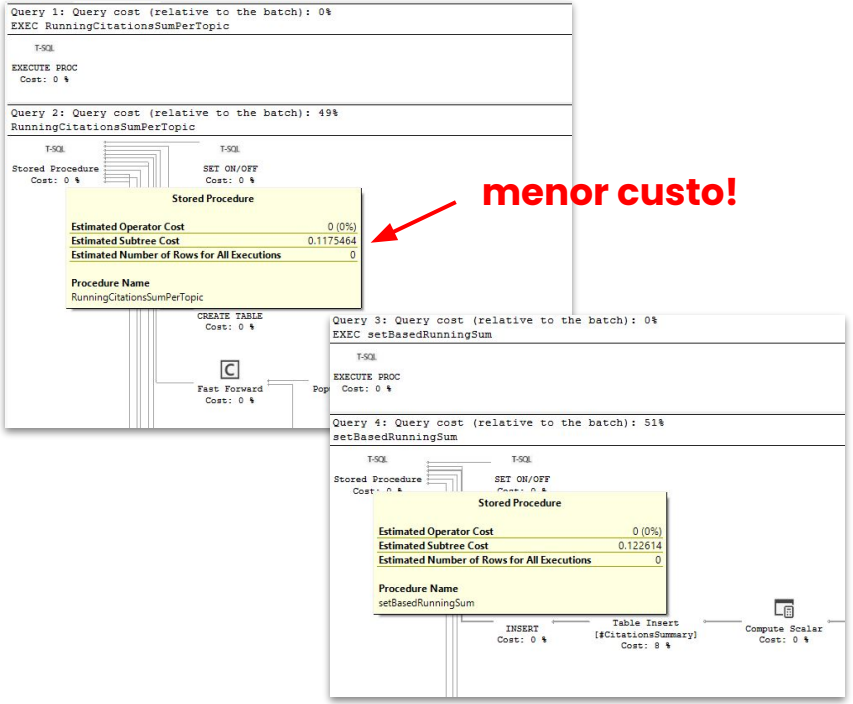
UDF!!

```
CREATE PROCEDURE Top3TopicsPerYear
AS
BEGIN
    SELECT PublicationYear, T.TopicName, TopicCount
    FROM (
        SELECT COUNT(L.TopicID) AS TopicCount, L.TopicName AS TopicName, RANK() OVER (PARTITION BY YEAR(PublicationDate)
ORDER BY COUNT(L.TopicID) DESC, L.TopicName) AS rank_pub, YEAR(PublicationDate) AS PublicationYear
        FROM ListAllArticlesPerTopic() AS L
        INNER JOIN JournalVolume ON JournalVolume.JournalID = L.JournalID AND JournalVolume.Volume = L.Volume
        GROUP BY L.TopicName, YEAR(PublicationDate)
    ) AS T
    WHERE T.rank_pub <= 3
    ORDER BY T.PublicationYear DESC, T.rank_pub
END;
```

Estatísticas - Cursor

```
(...)

WHILE @@FETCH_STATUS = 0
BEGIN
    -- Add the current CitationsCount to the running sum
    SET @RunningSum = @RunningSum + @CitationsCount

    -- Insert the current row's data and the running sum into the temporary table
    INSERT INTO #CitationsSummary (TopicName, CitationsCount, RunningCitationsSum)
    VALUES (@TopicName, @CitationsCount, @RunningSum)

    -- Fetch the next row from the cursor
    FETCH NEXT FROM runningSumCursor INTO @TopicName, @CitationsCount
END
(...)
```

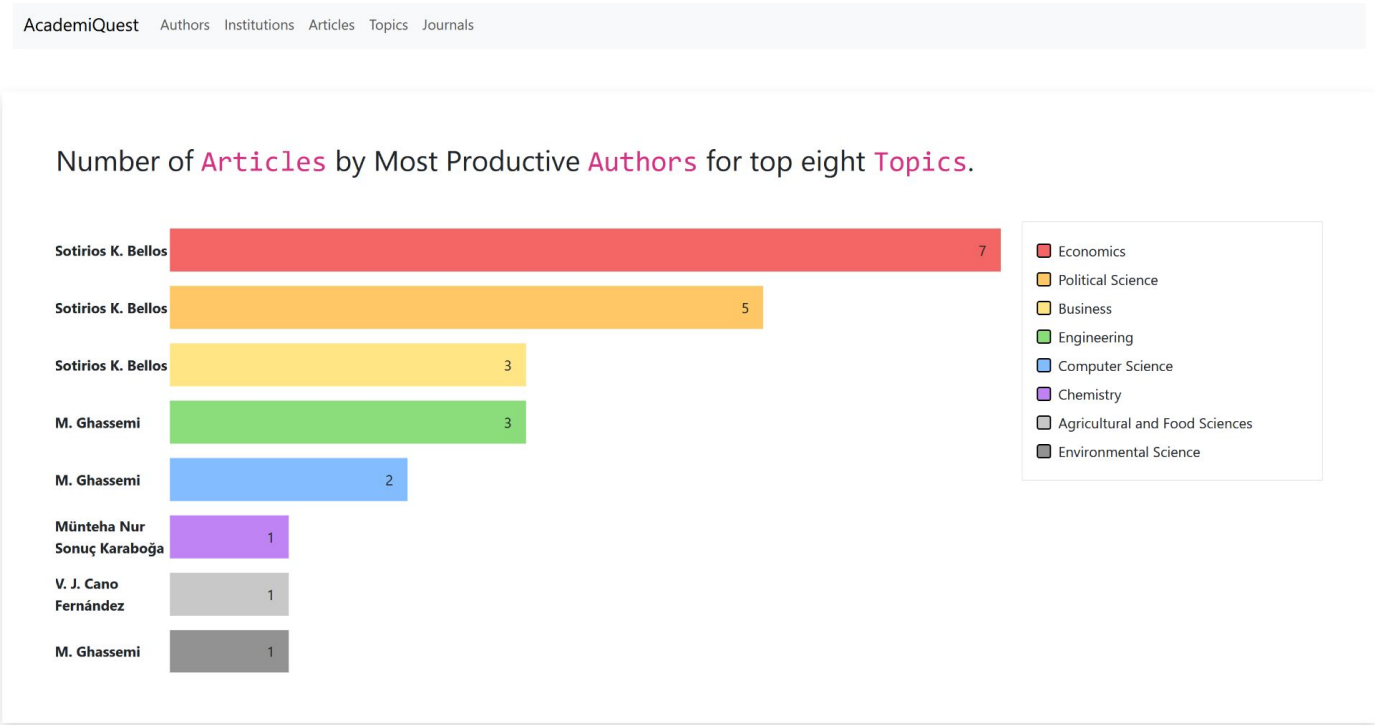




menor custo!

Soluções Set-based são mais lentas neste tipo de requisitos

Vídeo Demonstrativo



(video)

deti

BD – Apresentação Final

AcademiQuest

Questões?

Grupo p5g1:
Pedro Pinto - 115304
João Pinto - 104384