# A Modifying Operator for Temporal Intuitionistic Fuzzy Index Matrices with Application to Autonomous Self-Managing Databases


**Veselina Bureva[1], Krassimir Atanassov[2]**

[1] Laboratory of Intelligent Systems,
Burgas State University "Prof. Dr. Assen Zlatarov",
"Prof. Yakimov" Blvd., Burgas 8010, Bulgaria
veselina-bureva@uniburgas.bg

[2] Department of Bioinformatics and Mathematical Modelling
Institute of Biophysics and Biomedical Engineering
Bulgarian Academy of Sciences
Acad. G. Bonchev Str., bl. 105, Sofia 1113, Bulgaria
k.t.atanassov@gmail.com



*20th International Workshop on Intuitionistic Fuzzy Sets, 12 December 2025*

# A Modifying Operator for Temporal Intuitionistic Fuzzy Index Matrices with Application to Autonomous Self-Managing Databases

## Brief Remarks on Index Matrices (IMs)

Let ***I*** be a fixed set of indices and ***R*** be the set of all real numbers. By IM with index sets ***K*** and ***L*** ($K, L \in I$), we mean the object

$$[K, L, \{a_{k_i,l_j}\}] \equiv \begin{array}{c|cccc} & l_1 & l_2 & \dots & l_n \\ \hline k_1 & a_{k_1,l_1} & a_{k_1,l_2} & \dots & a_{k_1,l_n} \\ k_2 & a_{k_2,l_1} & a_{k_2,l_2} & \dots & a_{k_2,l_n} \\ \vdots & \dots & \dots & \dots & \dots \\ k_m & a_{k_m,l_1} & a_{k_m,l_2} & \dots & a_{k_m,l_n} \end{array}$$

where $K = \{k_1, k_2, ..., k_m\}$, $L = \{l_1, l_2, ..., l_n\}$, and for $1 \le i \le m$, and $1 \le j \le n : a_{k_i,l_j} \in R$

The theory of IMs is described in "Atanassov, K. (2014). *Index Matrices: Towards an Augmented Matrix Calculus*. *Studies in Computational Intelligence Series, Vol. 573, Springer, Cham.*".

Therefore, the types of index matrices extensions include **intuitionistic fuzzy index matrices** (IFIM), ***n*-dimensional intuitionistic fuzzy index matrices** (*n*-DIFIM), **Index matrices with function-type elements** (IMFE)**, temporal intuitioninistic fuzzy index matrices** (TIFIM).

# A Modifying Operator for Temporal Intuitionistic Fuzzy Index Matrices with Application to Autonomous Self-Managing Databases

In the current investigation, we will use TIFIMs. Each TIFIM has **three dimensions** assigned with letters ***K*, *L* and *T***, where ***K* and *L* are sets of indices** and ***T* is some fixed temporal scale**. The TIFIM has the following form:

$$A(T) = \left[K, L, T, \langle \mu_{k_i,l_j,\tau}, \nu_{k_i,l_j,\tau} \rangle\right]$$

$$\equiv \left\{ \begin{array}{c|ccccc} \tau & l_1 & \dots & l_j & \dots & l_n \\ \hline k_1 & \langle \mu_{k_1,l_1,\tau}, \nu_{k_1,l_1,\tau} \rangle & \dots & \langle \mu_{k_1,l_j,\tau}, \nu_{k_1,l_j,\tau} \rangle & \dots & \langle \mu_{k_1,l_n,\tau}, \nu_{k_1,l_n,\tau} \rangle \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ k_i & \langle \mu_{k_i,l_1,\tau}, \nu_{k_i,l_1,\tau} \rangle & \dots & \langle \mu_{k_i,l_j,\tau}, \nu_{k_i,l_j,\tau} \rangle & \dots & \langle \mu_{k_i,l_n,\tau}, \nu_{k_i,l_n,\tau} \rangle \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ k_m & \langle \mu_{k_m,l_1,\tau}, \nu_{k_m,l_1,\tau} \rangle & \dots & \langle \mu_{k_m,l_j,\tau}, \nu_{k_m,l_j,\tau} \rangle & \dots & \langle \mu_{k_m,l_n,\tau}, \nu_{k_m,l_n,\tau} \rangle \end{array} \middle| \tau \in T \right\}$$

where $\boldsymbol{\tau}$ **is an element of *T***, i.e., a time-moment for every $\tau \in T$, $1 \leq i \leq m, 1 \leq j \leq n$:

$$\mu_{k_i,l_j,\tau}, \nu_{k_i,l_j,\tau}, \mu_{k_i,l_j,\tau} + \nu_{k_i,l_j,\tau} \in [0,1].$$

# A Modifying Operator for Temporal Intuitionistic Fuzzy Index Matrices with Application to Autonomous Self-Managing Databases

Let us have the **time-moments** $\tau_1$ **and** $\tau_2$ **such that** $\{\tau_1, \tau_2\} \in T$ and let us have **the constants** $c$, $d \in [0,1]$ **such that** $c + d \leq 1$. Therefore, we define following **modifying operator** for TIFIM:

$$M(A(T)) = \left[K, L, T, \langle \rho_{k_i,l_j,\tau}, \sigma_{k_i,l_j,\tau} \rangle\right],$$

where

$$\langle \rho_{k_i,l_j,\tau}, \sigma_{k_i,l_j,\tau} \rangle =$$

$$= \begin{cases} \langle \mu_{k_i,l_j,\tau} + c.\dfrac{\tau_1 - \tau}{\tau_1}.\nu_{k_i,l_j,\tau}, d.\dfrac{\tau}{\tau_1}.\nu_{k_i,l_j,\tau} \rangle, & if\ k_i \in K, l_j \in L, \tau \in T\ and\ \tau \leq \tau_1 \\ \langle c.\dfrac{\tau_2 - \tau}{\tau_2 - \tau_1}.\mu_{k_i,l_j,\tau}, \nu_{k_i,l_j,\tau} + d.\dfrac{\tau - \tau_1}{\tau_2 - \tau_1}.\mu_{k_i,l_j,\tau} \rangle, & if\ k_i \in K, l_j \in L, \tau \in T\ and\ \tau_1 < \tau < \tau_2 \\ \langle 0,1 \rangle, & if\ k_i \in K, l_j \in L, \tau \in T\ nd\ \tau \geq \tau_2 \end{cases}$$

# A Modifying Operator for Temporal Intuitionistic Fuzzy Index Matrices with Application to Autonomous Self-Managing Databases

In the current investigation, the **searching data frequency** is observed. The datasets records **priorities are calculated**. Depending on the appropriate time-moments and the searching frequency the **database states are updated**.

The frequency estimation is defined using the notation of intuitionistic fuzzy sets. Similar investigation for intuitionistic fuzzy evaluations calculation while performing INSERT and UPDATE statements are discussed in „*Bureva, V., Petrov, P., Andonov, V., Atanassov, K. (2023). Intuitionistic Fuzzy Evaluation of User Requests Frequency. In: Atanassov, K.T., et al. Uncertainty and Imprecision in Decision Making and Decision Support - New Advances, Challenges, and Perspectives. IWIFSGN BOS/SOR 2022 2022. Lecture Notes in Networks and Systems, vol 793. Springer, Cham., pp 15-2*“

In the next example the knowledge from the theory of expert systems, the new modifying operator and databases are integrated to develop an example for self-management system.

# A Modifying Operator for Temporal Intuitionistic Fuzzy Index Matrices with Application to Autonomous Self-Managing Databases

## 2. Autonomous Priority-Based Self-Managing Database System

The presented investigation describes an "Autonomous Priority-Based Self-Managing Database System". The system **dynamically adjusts the priority of stored records based on their search frequency**. It:

– automatically **reduces** the priority of rarely accessed data,

– **deletes** outdated information,

– and **reinforces** the importance of frequently selected records.

Through scheduled maintenance and adaptive control functions, the database maintains optimal relevance and efficiency without human intervention. The records priorities are presented in the form of intuitionistic fuzzy pairs. Therefore, an example of modifying operator over TIFIM is presented.

# A Modifying Operator for Temporal Intuitionistic Fuzzy Index Matrices with Application to Autonomous Self-Managing Databases

The TIFIM is represented as data table. The database is implemented using *PostgreSQL* database management system. An array data type is used for the “*priority*” column. Additional rule to the “*priority*” column is assigned: its values and their sum have to be in the interval [0, 1]. The column *last_search* contains the information for the data last search.

Query  Query History

```sql
CREATE DATABASE DataItemsSearch;

CREATE TABLE dataItems (
    item_id SERIAL PRIMARY KEY,
    content TEXT NOT NULL,
    priority NUMERIC[2] NOT NULL DEFAULT '{0,1}',
    last_search TIMESTAMP,
    CHECK (
        priority[1] >= 0 AND priority[1] <= 1 AND
        priority[2] >= 0 AND priority[2] <= 1 AND
        (priority[1] + priority[2]) <= 1
    )
);
```

Fig.1 SQL statement for *dataItems* table

# A Modifying Operator for Temporal Intuitionistic Fuzzy Index Matrices with Application to Autonomous Self-Managing Databases

The records into the *dataItems* table are inserted. The priority field contains the intuitionistic fuzzy pairs. They are calculated according to the records searching frequency. In each time moment, the table has different state depending on the priority field. The values are calculated dynamically.

| item_id [PK] integer | content text | priority numeric[] | last_search timestamp without time zone |
|---|---|---|---|
| 1 | Python tutorial | {0.8,0.2} | 2025-11-10 12:00:00 |
| 2 | SQL joins | {0.7,0.2} | 2025-11-09 08:00:00 |
| 3 | JavaScript intro | {0.6,0.3} | 2025-10-01 09:00:00 |
| 4 | Medical Information Systems | {0.0,1.0} | 2025-11-10 12:00:00 |
| 5 | Diseases | {1.0,0.0} | 2025-11-09 08:00:00 |
| 6 | Healthcare Report | {0.5,0.4} | 2025-10-01 09:00:00 |
| 7 | Artificial Intelligence | {0.8,0.2} | 2025-11-25 12:00:00 |

Fig.2 State of the *dataItems* table for time moment *t*

# A Modifying Operator for Temporal Intuitionistic Fuzzy Index Matrices with Application to Autonomous Self-Managing Databases

The first step of the autonomous priority-based self-managing database system development is to provide the methodology for priority calculation. It has the following form:

- If the record is **frequently searched**, the degree of membership increases and the degree of non-membership decreases;
- If the record is **infrequently, rarely searched** (between $x$ and $y$ days), the degree of membership decreases and the degree of non-membership increases;
- If the record is **non searched more than *n* days** then the row is deleted after setting its priority to $\langle 0,1 \rangle$

Therefore, the next logic step is to define four functions:

- *update_priority_on_interval1*() – **increase the priority** when the user searches in the row;
- *update_priority_on_interval2*() – **decrease priority** for inactive records;
- *update_priority_on_interval3*()– **setting values ⟨0, 1⟩ to old records** that are not searched $n$ days.
- *delete_old_data*() – **delete old records** that are not searched $n$ days.

# A Modifying Operator for Temporal Intuitionistic Fuzzy Index Matrices with Application to Autonomous Self-Managing Databases

The *first* function changes the priority of the frequently selected records. The ***searched text, the constants c and d and time interval have*** to be assigned in the function-calling step.

In the presented example a word "*Artificial Intelligence*" searching is performed. The value 0.1 and 0.3 are assigned to the constants *c* and *d*. After the function execution the priority of the record with id=7.

Query | Query History

```sql
CREATE OR REPLACE FUNCTION update_priority_interval1(
    search_term TEXT,
    c NUMERIC,
    d NUMERIC,
    tau1 INTERVAL
)
RETURNS VOID AS $$
BEGIN
    UPDATE dataitems
    SET
        priority = ARRAY[
            LEAST(priority[1] + c * ((EXTRACT(EPOCH FROM (tau1 - (NOW() - last_search))) / EXTRACT(EPOCH FROM tau1))) * priority[2], 1),
            GREATEST(d * ((EXTRACT(EPOCH FROM (NOW() - last_search)) / EXTRACT(EPOCH FROM tau1))) * priority[2], 0)
        ],
        last_search = CASE WHEN search_term <> '%' THEN NOW() ELSE last_search END
    WHERE content ILIKE '%' || search_term || '%'
      AND NOW() - last_search <= tau1;
END;
$$ LANGUAGE plpgsql;

SELECT update_priority_interval1('Artificial Intelligence', 0.1, 0.3, '7 days'::interval);
```

Data Output | Messages | Notifications

| | item_id [PK] integer | content text | priority numeric[] | last_search timestamp without time zone |
|---|---|---|---|---|
| 1 | 1 | Python tutorial | {0.8,0.2} | 2025-11-10 12:00:00 |
| 2 | 2 | SQL joins | {0.7,0.2} | 2025-11-09 08:00:00 |
| 3 | 3 | JavaScript intro | {0.6,0.3} | 2025-10-01 09:00:00 |
| 4 | 4 | Medical Information Systems | {0.0,1.0} | 2025-11-10 12:00:00 |
| 5 | 5 | Diseases | {1.0,0.0} | 2025-11-09 08:00:00 |
| 6 | 6 | Healthcare Report | {0.5,0.4} | 2025-10-01 09:00:00 |
| 7 | 7 | Artificial Intelligence | {0.81100172337291666666,0.02699482988125000002} | 2025-11-28 15:35:07.885203 |

Fig.3 Function that changes the priority of the recording according to the searching activities

# A Modifying Operator for Temporal Intuitionistic Fuzzy Index Matrices with Application to Autonomous Self-Managing Databases

The second function provides functionalities for **decreasing the priority of rarely searched items**. In the presented example, the **time interval between 7 and 30 days** is considered for the rare searched items determination.

Query | Query History

```sql
CREATE OR REPLACE FUNCTION update_priority_interval2(
    search_term TEXT,
    c NUMERIC,
    d NUMERIC,
    tau1 INTERVAL,
    tau2 INTERVAL
)
RETURNS VOID AS $$
BEGIN
    UPDATE dataitems
    SET
        priority = ARRAY[
            LEAST(GREATEST(c * ((EXTRACT(EPOCH FROM (tau2 - (NOW() - last_search))) / EXTRACT(EPOCH FROM (tau2 - tau1)))) * priority[1],0),1),
            LEAST(GREATEST(priority[2] + d * ((EXTRACT(EPOCH FROM ((NOW() - last_search) - tau1)) / EXTRACT(EPOCH FROM (tau2 - tau1)))) * priority[1],0),1)
        ]
    WHERE content ILIKE '%' || search_term || '%'
      AND NOW() - last_search > tau1
      AND NOW() - last_search < tau2;
END;
$$ LANGUAGE plpgsql;

SELECT update_priority_interval2('%', 0.1, 0.3, '7 days'::interval, '30 days'::interval);
SELECT  item_id, content, priority, last_search FROM dataitems;
```

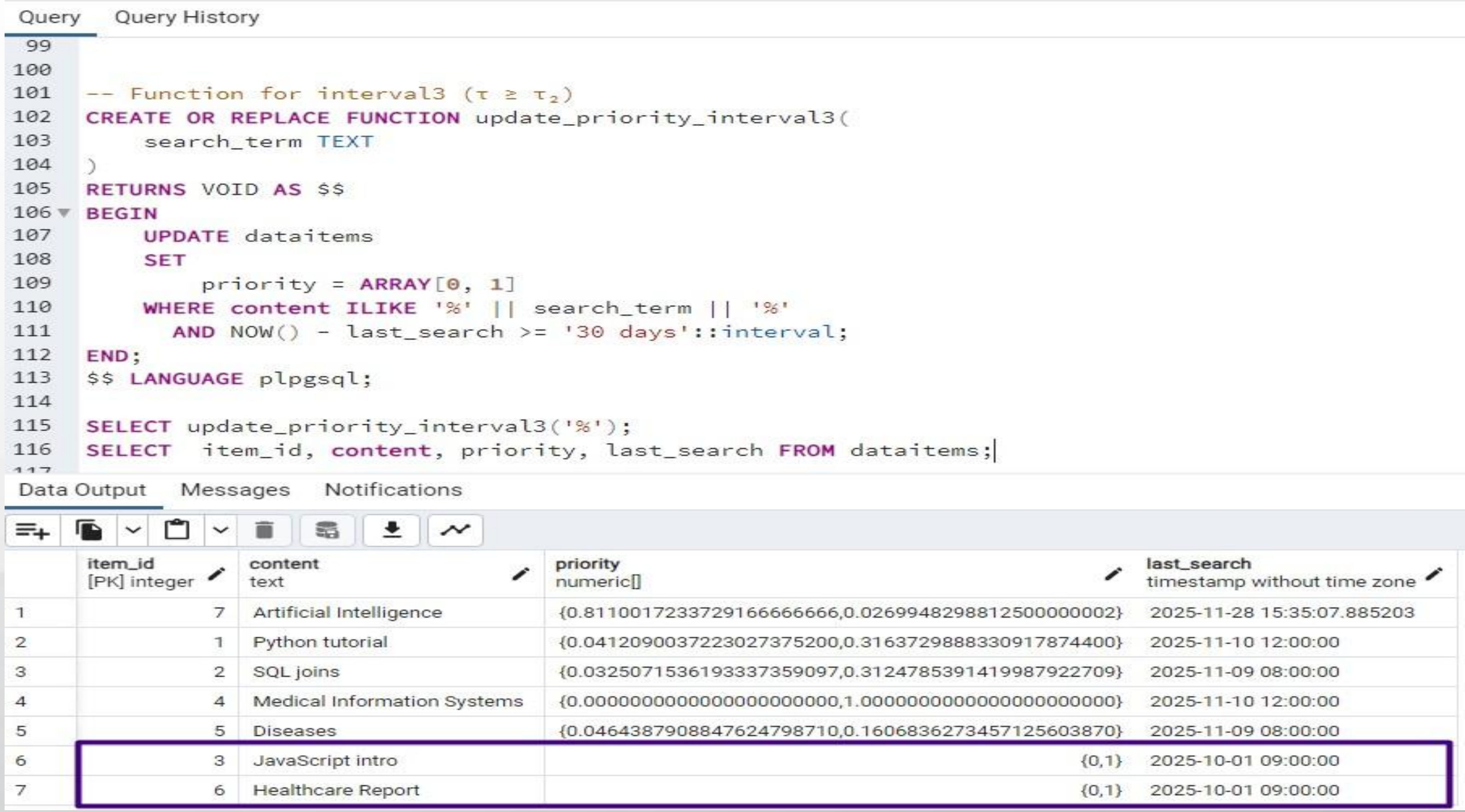
Data Output | Messages | Notifications

| | item_id [PK] integer | content text | priority numeric[] | last_search timestamp without time zone |
|---|---|---|---|---|
| 1 | 3 | JavaScript intro | {0.6,0.3} | 2025-10-01 09:00:00 |
| 2 | 6 | Healthcare Report | {0.5,0.4} | 2025-10-01 09:00:00 |
| 3 | 7 | Artificial Intelligence | {0.81100172337291666666,0.02699482988125000002} | 2025-11-28 15:35:07.885203 |
| 4 | 1 | Python tutorial | {0.0412090037223027375200,0.3163729888330917874400} | 2025-11-10 12:00:00 |
| 5 | 2 | SQL joins | {0.0325071536193337359097,0.3124785391419987922709} | 2025-11-09 08:00:00 |
| 6 | 4 | Medical Information Systems | {0.0000000000000000000000,1.0000000000000000000000} | 2025-11-10 12:00:00 |
| 7 | 5 | Diseases | {0.0464387908847624798710,0.1606836273457125603870} | 2025-11-09 08:00:00 |

Fig.4 Function that decrease the priority of the rarely searched items

# A Modifying Operator for Temporal Intuitionistic Fuzzy Index Matrices with Application to Autonomous Self-Managing Databases

In the third function has the capabilities for **setting the values ⟨0, 1⟩ to the records that are not searched more than 30 days**.




```sql
-- Function for interval3 (τ ≥ τ₂)
CREATE OR REPLACE FUNCTION update_priority_interval3(
    search_term TEXT
)
RETURNS VOID AS $$
BEGIN
    UPDATE dataitems
    SET
        priority = ARRAY[0, 1]
    WHERE content ILIKE '%' || search_term || '%'
      AND NOW() - last_search >= '30 days'::interval;
END;
$$ LANGUAGE plpgsql;

SELECT update_priority_interval3('%');
SELECT  item_id, content, priority, last_search FROM dataitems;
```

| | item_id [PK] integer | content text | priority numeric[] | last_search timestamp without time zone |
|---|---|---|---|---|
| 1 | 7 | Artificial Intelligence | {0.811001723372916666666,0.0269948298812500000002} | 2025-11-28 15:35:07.885203 |
| 2 | 1 | Python tutorial | {0.0412090037223027375200,0.3163729888330917874400} | 2025-11-10 12:00:00 |
| 3 | 2 | SQL joins | {0.0325071536193337359097,0.3124785391419987922709} | 2025-11-09 08:00:00 |
| 4 | 4 | Medical Information Systems | {0.0000000000000000000000,1.0000000000000000000000} | 2025-11-10 12:00:00 |
| 5 | 5 | Diseases | {0.0464387908847624798710,0.1606836273457125603870} | 2025-11-09 08:00:00 |
| 6 | 3 | JavaScript intro | {0,1} | 2025-10-01 09:00:00 |
| 7 | 6 | Healthcare Report | {0,1} | 2025-10-01 09:00:00 |

Fig.5 Function that deletes the non-searched items

# A Modifying Operator for Temporal Intuitionistic Fuzzy Index Matrices with Application to Autonomous Self-Managing Databases

The result of the *delete_old_data*() function execution is presented. The records that are not searched more than a month are **deleted** from the *dataItems* table. The new state of the table contains the frequently searched data according to the previously defined conditions.

Query | Query History

```sql
CREATE OR REPLACE FUNCTION delete_old_data()
RETURNS VOID AS $$
BEGIN
    DELETE FROM dataitems
    WHERE NOW() - last_search > INTERVAL '30 days';
END;
$$ LANGUAGE plpgsql;


SELECT delete_old_data()
SELECT  item_id, content, priority, last_search FROM dataitems;
```

Data Output | Messages | Notifications

| | item_id [PK] integer | content text | priority numeric[] | last_search timestamp without time zone |
|---|---|---|---|---|
| 1 | 7 | Artificial Intelligence | {0.81100172337291666666,0.02699482988125000002} | 2025-11-28 15:35:07.885203 |
| 2 | 1 | Python tutorial | {0.0412090037223027375200,0.3163729888330917874400} | 2025-11-10 12:00:00 |
| 3 | 2 | SQL joins | {0.0325071536193337359097,0.3124785391419987922709} | 2025-11-09 08:00:00 |
| 4 | 4 | Medical Information Systems | {0.00000000000000000000,1.00000000000000000000} | 2025-11-10 12:00:00 |
| 5 | 5 | Diseases | {0.0464387908847624798710,0.1606836273457125603870} | 2025-11-09 08:00:00 |

Fig.6 Result of the delete function execution

# A Modifying Operator for Temporal Intuitionistic Fuzzy Index Matrices with Application to Autonomous Self-Managing Databases

In the next step the system is set to be self-managing. The *update_priority_on_interval1*() is **rewritten as trigger function** that enables autonomous self-management in the database by capturing every user search request and automatically invoking the priority-update mechanism.

Upon insertion of a new search query into search_log, the function activates the priority recalculation algorithm.

Therefore, the system will continuously adapt item priorities based on real-time user behavior, without manual intervention.

Query    Query History

```sql


-- Trigger-function for current moment searching
CREATE OR REPLACE FUNCTION trg_update_priority_on_search()
RETURNS TRIGGER AS $$
BEGIN
    PERFORM update_priority_interval1(NEW.search_term, 0.1, 0.3, '7 days'::interval);
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

--  Trigger
CREATE TRIGGER trg_update_priority
AFTER INSERT ON search_log
FOR EACH ROW
EXECUTE FUNCTION trg_update_priority_on_search();
```

Fig.7 Trigger function *trg_update_priority_on_search*()

# A Modifying Operator for Temporal Intuitionistic Fuzzy Index Matrices with Application to Autonomous Self-Managing Databases

The tasks of decreasing priority and deleing records are atomized using *pg_cron* extension for PostgreSQL.

- The **first job** is used for **modifying priorities of the not searched data between 7 and 30 days**. The update will be every night at 2:00 pm.
- The **second job is used for setting the values ⟨0, 1⟩ to the records that are not searched more than 30 days**. It is activated every night at 2.05 pm.
- The **third job** will **delete the rarely searched data (>30 days)** every night at 3:00 pm.

Query    Query History

```
--  Cron tasks for automatic decay
CREATE EXTENSION IF NOT EXISTS pg_cron;

-- Decay second interval
SELECT cron.schedule(
    'daily_decay_interval2',
    '0 2 * * *',
    $$SELECT update_priority_interval2('%', 0.1, 0.3, '7 days'::interval, '30 days'::interval);$$
);

-- Decay third interval
SELECT cron.schedule(
    'daily_decay_interval3',
    '5 2 * * *',
    $$SELECT update_priority_interval3('%');$$
);

-- Delete old data
SELECT cron.schedule(
    'daily_cleanup',
    '0 3 * * *',
    $$SELECT delete_old_data();$$
);
```

Fig.8 Cron jobs

# A Modifying Operator for Temporal Intuitionistic Fuzzy Index Matrices with Application to Autonomous Self-Managing Databases

In the current investigation a **modifying operator over TIFIM** is discussed. An application in the field of **self-management databases** is presented. The **dynamic priorities** are implemented. The activities related to the modifying operator are automated in the autonomous database system.

# A Modifying Operator for Temporal Intuitionistic Fuzzy Index Matrices with Application to Autonomous Self-Managing Databases

This research was funded by the Bulgarian National Science Fund under the Grant KP-06-N72/8 from 14.12.2023 "Intuitionistic fuzzy methods for data analysis with an em-phasis on the blood donation system in Bulgaria".

# Thank you for your attention!