International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 September 2023*

Air Quality Data Acquisition Through Temporal Data Modeling

[1]Lalit Gandhi, [2]Rahul Rishi, [3]Sonia Sharma
[1]University Institute of Engineering and Technology
Maharshi Dayanand University
Rohtak - 124001, India
lalit.gandhi@mdurohtak.ac.in
[2]University Institute of Engineering and Technology
Maharshi Dayanand University
Rohtak - 124001, India
rahulrishi@mdurohtak.ac.in
[3]University Institute of Engineering and Technology
Maharshi Dayanand University
Rohtak - 124001, India
soniasharma.rp.uiet mdurohtak.ac.in

Abstract - The paper presents a novel temporal data model for acquiring air quality data. The proposed model uses the free and open-source Postgres database with tuple time stamping, resulting in rapid and effective data extraction. The conventional relational database SQL server for data extraction is used in this paper as the foundation for an air quality temporal database model. The air quality data set collected for MDU Rohtak (source: Central Control Room for Air Quality Management - All India) is utilized to test and determine the outcomes of the proposed model. Valid time is used as the time dimension for capturing data values, together with tuple timestamping. The Postgres "tsrange" datatype is used to record the timestamp. To eliminate data duplication, the dataset is cleaned up and normalized. Using the air quality temporal data model, AQI data at various time points is effectively extracted. The present model data extraction is compared with the conventional relational data model using SQL Server database. The air quality data extraction process takes a remarkably minimal amount of time, with an average of 23% lesser time than SQL server database. It takes only a few milliseconds to extract the AQI data, and the results of the experiment are ready immediately after that. The suggested model is helpful in decision-making and offers decision-makers useful information and the tool they need to enhance air quality. The results are relevant to the study of the AQI components by different pollution management agencies like central and state pollution control boards, national green tribunal and Ministry of the Environment for efficient air pollution control measures.

Keywords - Air quality index, temporal operators, data acquisition

I. INTRODUCTION

The continuous assessment of the air quality is done using the air quality index(AQI). The AQI is calculated using air pollutants concentration over a defined averaging period collected from an air monitor [1]. As the AQI levels rise, so do the health risks, so the primary goal of measuring the AQI is to educate people about how air quality affects their health. Continuous monitoring of air quality parameters like PM2.5, CO, SO_2, NO_2, benzene etc. help authorities to take necessary action to control the air pollution. Recent studies have revealed that there is a definite correlation between population exposure to air pollutants and their harmful effects on health. The air quality index serves to assess air quality by continually monitoring primary harmful emissions such as ground-level ozone (O_3), carbon monoxide (CO), Sulphur dioxide (SO_2), nitrogen dioxide (NO_2), and particle pollution/particulate matter PM2.5 [1-2]. Patients can use the air quality health index to assess risk before engaging in outdoor activity and reduce pollution's effects is to reduce exposure, particularly from traffic. Several agencies, like the central as well state pollution control board and the Ministry of Environment, use the information obtained to establish effective air pollution prevention policies.

Time is a critical component in continuously monitoring and analyzing data for air quality, and temporal data models are the top pick for implementation [3]. Numerous academicians and researchers have developed specialized techniques in temporal relational databases [3-15]. The temporal data model highlights semantics, including query processing, query storage, and implementation. Temporal query processing is the processing of temporal data and the

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 September 2023*

implementation of query execution plans. A temporal database contains information about specific points in time. [4-5]. A literature review of various temporal data models is carried out here to prepare for the presentation of an air quality temporal data model. Temporal Functionality in Objects with Roles Mode (TF-ORM) is a temporal system for database management that is layered on top of a traditional database [6], Temporal Relational Model for Managing Patient data (TempR-PDM) is a proposed framework of temporal relations for handling patients data [7-10], Temporal data management–BCDM [11], and several other data models including TERADATA, tuple timestamped single relation. These models are quite straightforward to implement and very effective regarding query execution time [12-20]. The focus of the work is on implementing temporal database characteristics, none of the researchers reported air quality temporal data modeling.

Therefore, to get air quality components data, this research paper suggests a temporal database model. The conventional relational data model that could hold only current values, served as the framework for the invention of temporal data models as needed to store previous and forecast values in the database as well. The suggested model's basis is tuple timestamping, and its output is in 1NF. Valid-time is taken into account when doing time stamping. Valid time is the time period during which a database fact is useful in real world. To record proper time, valid-from and valid-to-time dimensions are added to the database table. The suggested temporal model for air quality can be utilized to access & assess and alter data on air quality while keeping track of tuple timestamps. The air quality data set collected for MDU Rohtak for the year 2020 (source: Central Control Room (CCR) for Air Quality Management - All India) is utilized to test and determine the outcomes of the proposed model using open-source DBMS Postgres version 10. The projected model utilizes the data and uses DBMS to extract the details at any particular day, or specific AQI components at special time or to get the trends of parts for range of time.

This paper is organized as follows: Section II describes the methodologies related to the logical design of the Temporal Database model. This section also includes Temporal Relational Algebra and a brief description of Postgres. Section III represents the proposed Air Quality Acquisition Temporal data model for air quality data retrieval. This section explains the architecture of the proposed model and schema representation with the range of operators applied for present study. Section IV contains the experimental setup, query formation and results. Finally, Section V concludes and points out guidelines for future research.

II. METHODOLOGIES OF TEMPORAL DATABASE

A data model is a representation of data elements and relationships between them. On a similar basis, the temporal data model represents the data elements along with time and relationship between them. Preserving each new set of updates is referred to as a "revision". Temporal Model enables to keep track of changes made to these entities throughout time. The ability to obtain an entity's state at any moment is made feasible by the fact that each revision is maintained along with time (temporal) data. Instead of using separate "log" columns, this temporal model approach saves entity revisions in the same table as the original data.

In temporal data modeling, time points are associated with the temporal relational schema R_t (attributes, start time, end time). More importantly, constraints related to the schema must also be met, such as the requirement that the start time be smaller than the end time or that the attributes not be NULL. Following is the representation of temporal relational structure

$$Relation_{time} = (Attr_1, Attr_2, Attr_3....Attr_n \mid Time_v) \quad \text{--Equation (1)}$$

Where $Relation_{time}$ is the temporal relation schema with a finite set of attributes $Attr_1$, $Attr_2$, $Attr_3$.... $Attr_n$ with valid time. $Domain_i$ represents a domain of attributes for $1 \leq i \leq n$. Relational schema $Relation_{time}$ has a relation (r) which is represented as r ($Relation_{time}$) and is a collection of mappings from $Relation_{time}$ to Domain.

Operators that apply on temporal relations and result in temporal relations are called temporal operators. These operators perform temporal transactions. Temporal operators are unary and binary in nature and operate upon single or double temporal relations. A unary method called "temporal selection" is used to retrieve information about time attributes that have been specified in the selection predicate P. Mathematically, temporal selection operator is defined as:

$$\sigma_P^t(relation) = \{t \mid t \grave{o}\ relation\ \Lambda\ P(t[A])\} \quad \text{--Equation (2)}$$

where σ^t , represents temporal selection, relation is the temporal relation, A is the set of attributes and P stands for propositional logic formula.

Temporal Union is a binary operator used to perform binary union between two compatible temporal relations and is denoted by $\cup_t$. When two relations share the same arity and attribute domain, they are said to be compatible. It can be represented as

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 September 2023*

$$r_1 \cup_t r_2 = \{t \mid \left(\exists x \in r_1\ \exists y \in r_2 \left(x[A] = y[A]\ \Lambda\ t[A] = x[A]\ \Lambda\ t[T] = x[T] \cup y[T]\right)\right)\ V$$

$$\left(\exists x \in r_1 (t[A] = x[A] \Lambda (\sim \exists y \in r_2 \left(y[A] = x[A]\right))\ \Lambda\ t[T] = x[T]\ \right)\ V$$

$$\left(\exists y \in r_2 (t[A] = y[A] \Lambda (\sim \exists x \in r_1 \left(x[A] = y[A]\right))\ \Lambda\ t[T] = y[T]\ \right)\} \quad \text{--Equation (3)}$$

There are three parts in the above expression, the first part finds common tuples from both relations, second part selects tuples from first relation and the third selects tuples from the second relation only. All the selected tuples are then combined based on the primary key to get the result. Tuples of relations r_1 *and* r_2 satisfying on a critical attribute are collapsed into a single tuple. The relations r1 and r2 are either temporal database relations or temporal relational result sets known as temporary relations.

To connect tuples from two temporal relations that share at least one characteristic, perform the binary operation known as a "temporal join". In addition, the attributes must have the same name and domain. Mathematically, the temporal join can be defined as:

$$r_1 \bowtie_t r_2 = \{t \mid \exists x \in r_1\, \exists y \in r_2\ (x[A] = y[A] \Lambda\, x[T] \cap y[T] \neq 0 \Lambda$$
$$t[A] = x[A] \Lambda\, t[B] = x[B] \Lambda\, t[C] = y[C]\ \Lambda\ \ t[T] = x[T] \cap y[T])\} \quad \text{--Equation (4)}$$

Where $\bowtie_t$ stands for Temporal Natural Join, r_1 and r_2 are either temporal database relations or temporal relational result set known as temporary relations.

Finding tuples that are available in one relation but absent from the second compatible relation is done using the binary operation known as the "temporal difference operation". Mathematically, temporal difference operation is described as:

$$r_1 -_t r_2 = \{t \mid \exists x \in r_1\ ((t[A] = x[A]) \Lambda$$
$$((\exists y \in r_2\ (t[A] = y[A]\ \Lambda\ t[T] = x[T] - y[T]))\ V$$
$$(\sim \exists y \in r_2\ (t[A] = y[A]\ \Lambda\ t[T] = x[T])))\} \quad \text{--Equation (5)}$$

Temporal Intersection is a binary operator used to perform binary intersection between two compatible temporal relations and is denoted by $\cap_t$. Two relations are said to be compatible if they are of same arity and domain of their attributes is same. Mathematically, temporal intersection operation is defined as:

$$r_1 \cap_t r_2 = r_1 -_t (r_1 -_t r_2), \quad \text{--Equation (6)}$$

where $\cap_t$ stands for Temporal set difference, r_1 and r_2 are either temporal database relations or relation result set known as temporary relations.

Query language is the most important part for a data model. Temporal modeling uses temporal query to extract the temporal data from the database at a particular instance of time [40]. For retrieving and modifying temporal data, temporal query language is applied. Postgres is the strong open source database, which provides a number of influential features to cater the need for temporal data management. PostgreSQL is the ideal option for developing and implementing a temporal data model because of its comprehensive backend support, highly parallelized, security, robust connectivity support, and customer-friendly interface. Time interval data, time type and time zone flexibility are distinct features of PostgreSQL.

III PROPOSED AIR QUALITY ACQUISITION TEMPORAL MODEL

Air quality temporal data model is proposed for retrieving and manipulating air quality temporal data. This proposed model is built upon tuple timestamping and the results are in 1NF [41, 42]. Valid time is considered as the time dimension to perform the time stamping.

A. ARCHITECTURE OF THE PROPOSED MODEL

Figure 1 shows the architecture of the suggested temporal data model for air quality is shown in Figure 1. Air quality temporal database stores data in temporal databases and DBMS with the temporal extensions is utilized for managing the database. Using an application program, users can retrieve the temporal data and feed input to extract the desired output. Users only have access to the abstract level; they are not shown the details of the temporal features. The two primary elements of the temporal data model are (1)

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 September 2023*

temporal data types that are used in the form of the domain for temporal characteristics and (2) temporal operators employed to carry out temporal functionality. PostgreSQL version 10 is used to implement the temporal data model. The timestamp with a pair of discrete times is stored in PostgreSQL using the "tsrange" data type. The query language of the suggested model uses a number of range operators, which are listed in Table 1.

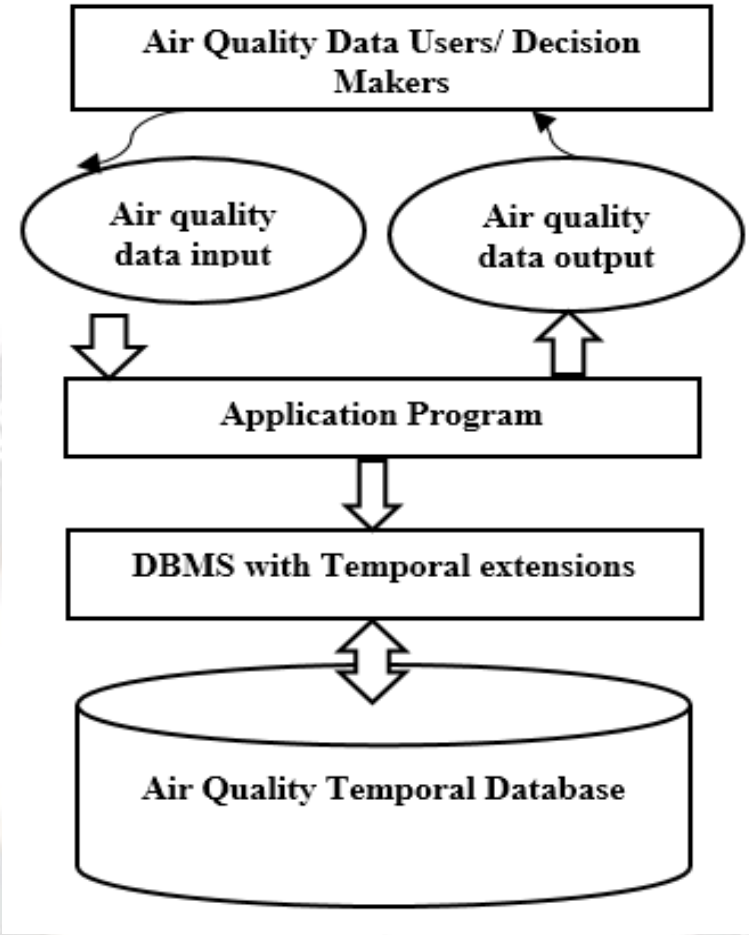


FIGURE 1 ARCHITECTURE OF AIR QUALITY INDEX DATA MODEL

B. SCHEMA REPRESENTATION:

The common air pollutants are PM2.5, nitrogen oxides and families like nitrogen oxide, nitrogen dioxide and sulphur dioxide, carbon monoxide that are recorded for every 15 minutes. Other hazardous pollutants are ozone, benzene, toluene, eth_benzene which are recorded on hourly basis along with temperature and rhesus (RH) factor. schema of the air quality data is represented as follows. There are two entity sets namely aqi_ per_15_mins and aqi_ per _hour. Schema of the air quality database is shown in figure 2 and represented as –

AQI_PER_15_MINS = (PM2.5, NO, NO2, NOX, SO2, CO, TIME_RANGE)

AQI_PER_HOUR = (OZONE, BENZENE, TOLUENE, ETH-BENZENE, TEMP, RH, TIME_RANGE)

Air quality attributes values of schema aqi_per_15_mins are recorded per fifteen minutes. Attributes values for aqi_per_hour are stored on hourly basis. Data type “tsrange” is used to store time_range attribute value in yyyymmdd hh:mm:ss format. Attribute values for "time_range” uniquely identifies the tuple and hence acts as the primary key.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 September 2023*

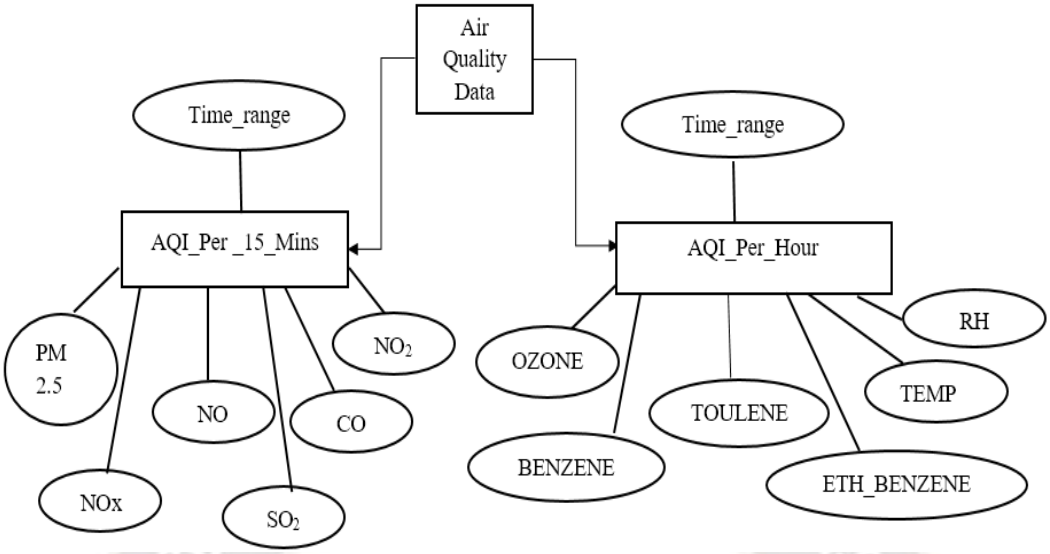


FIGURE 2: SCHEMA OF AIR QUALITY DATA

TABLE 1: RANGE OPERATORS USED FOR PRESENT STUDY

Sr. No.	Symbol	Operator
1	@>	Represents it contains element
2	&&	Represents Overlapping
3	>>	Represents strictly to the right of
4	<<	Represents strictly to the left of
5	&<	Represents it does not extend to the right of
6	&>	Represents it does not extend to left of

IV. EXPERIMENTAL SETUP, QUERY FORMATION & RESULTS

On an Intel(r) core (tm) i7-3770 cpu running at 3.40 GHz with 8GB of ram and windows 10 as the operating system, all experiments are conducted. Postgre SQL version 10 is the underlying database. Air quality data set gathered is used to test and find results using the model proposed. The data set collected from the central control room site is processed as shown in figure 3. The dataset received from the central control room site in excel format is preprocessed and converted to comma separated values format. Time range component "tsrange"- range of timestamp is added so that data can be recorded in postgre SQL format. "tsrange" datatype stores valid time range that is valid_ from and valid to time values. . "tsrange" data type is used to store time_range attribute value in yyyymmdd hh:mm:ss format. The main relation is decomposed to remove the duplicity of the records. There are two first normal form relations aqi_per_15_mins and aqi_per_hour with 35041 and 8760 records respectively. Postgre SQL version 10 is used for database creation and time stamping the records with tsrange. The experimental findings show that all air quality components for any date may be acquired in milliseconds. With the use of temporal data modelling, trends for each given aqi component may be found in just a few milliseconds. United states environmental protection agency (usepa) and uk- department of environment food and rural affairs (defra) publish a daily air quality index (daqi), british columbia aqhi (air quality health index) are some international agencies that uses aqi.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 September 2023*

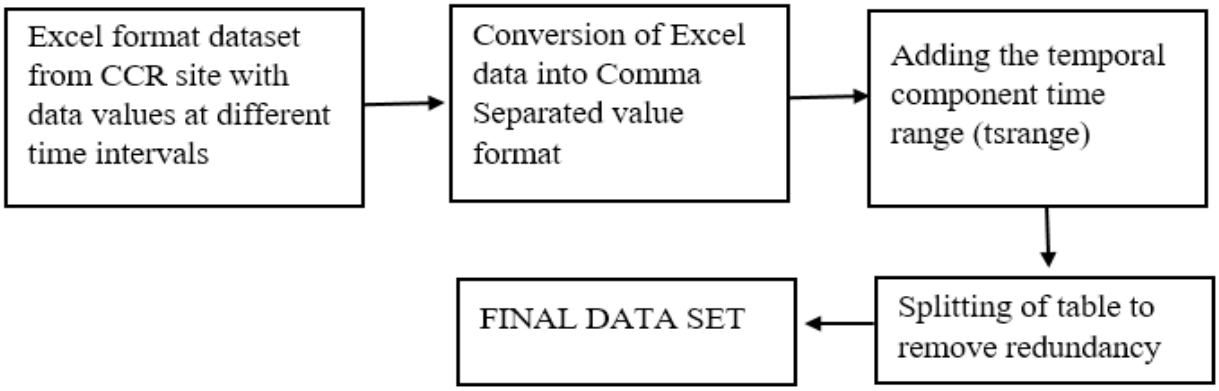


FIGURE 3. DATA PROCESSING FLOW CHART

A few queries have been created to study AQI temporal database model.

1. Show all attributes of the air quality records till 12 noon of 15th November 2020, one day after Diwali i.e.,14th November 2020.
2. Query to fetch Air quality data details when the PM2.5 lies between 100 and 200 µg/m3.
3. Query to generate air quality data details at a particular date and time
4. Show the trend of Benzene in the month of October 2020 using graphs.
5. Show the variations in temperature in the month of June 2020 using graphs.

4.1 RESULTS AND DISCUSSION

This section performs different temporal queries on the temporal database.

Query 1: Show all the air quality records till 12 noon of 15th November 2020, one day after Diwali -14th November. Query is well chosen to fetch the air quality data details one day after Diwali -14th November 2020, because air quality turns "severe" because of toxic smog after Diwali. The queries are given below, results from PostgreSQL 10 database is shown in figure 4 and graphical representation is shown in figure 5.

Query Code using SQL Server

```
SELECT * FROM AQI_PER_15_MINS A JOIN AQI_PER_HOUR B ON CAST (A.START_TIME AS DATE) = CAST (B.START_TIME AS DATE) AND DATEPART (HOUR, A.START_TIME) =DATEPART (HOUR, B.START_TIME) WHERE A.START_TIME >= '2020-11-15 00:00:00.000' AND A.END_TIME <= '2020-11-15 11:00:00.000';
```

It took 63 milliseconds to complete the extraction of data using SQL server with non-temporal database

Query Code using Postgres with Temporal Data Modeling

```
SELECT AQI.TIMERANGE, AQI."PM25", AQI."NO", AQI."NOX", AQI."NO2", AQI."SO2", AQI."CO", BENZENE, TOLUENE, OZONE, ETH_BENZENE, TEMP, RH FROM
(SELECT * FROM PUBLIC."AQI_PER_15_MINS" WHERE PUBLIC."AQI_PER_15_MINS".TIMERANGE && TSRANGE ('2020-11-15 00:00:00', '2020-11-15 11:00:00')) AQI
INNER JOIN
(SELECT * FROM PUBLIC."AQI_PER_HOUR" WHERE PUBLIC."AQI_PER_HOUR".TIMERANGE && TSRANGE ('2020-11-15 00:00:00', '2020-11-15 11:00:00')) AQI_H
ON AQI.TIMERANGE && AQI_H.TIMERANGE ORDER BY AQI.TIMERANGE;
```

The results of both queries are examined, and it is concluded that data can be efficiently acquired with temporal data modeling. Only 47 milliseconds are required to complete the optimal extraction using the temporal data modeling whereas conventional relational data modeling taken 63 milliseconds to complete the extraction. The query performance is significant, and the data extraction is accelerated. Figure 5 shows the mean value of all attributes as well as the percentage of constituents.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 September 2023*

	timerange tsrange	PM25 real	NO real	NOX real	NO2 real	SO2 real	CO real	benzene real	toulene real	eth_benzene real	temp real	rh real
1	["202 Read-only column 20-11-15 00:15:00")	338.16	8.09	23.74	42.13	17.64	[null]	45.41	6.83	11.79	15.79	59.55
2	["2020-11-15 00:15:00","2020-11-15 00:30:00")	338.16	4.49	30.01	57.44	3.23	1.63	45.41	6.83	11.79	15.79	59.55
3	["2020-11-15 00:30:00","2020-11-15 00:45:00")	265.17	6.08	41.17	76.32	1.63	1.63	45.41	6.83	11.79	15.79	59.55
4	["2020-11-15 00:45:00","2020-11-15 01:00:00")	253.94	5.44	46.43	87.36	1.04	1.81	45.41	6.83	11.79	15.79	59.55
5	["2020-11-15 01:00:00","2020-11-15 01:15:00")	253.94	7.41	45.19	81.95	13.19	1.77	52	6.87	11.38	15.65	59.62
6	["2020-11-15 01:15:00","2020-11-15 01:30:00")	253.94	8.62	45.88	81.38	12.27	1.86	52	6.87	11.38	15.65	59.62
7	["2020-11-15 01:30:00","2020-11-15 01:45:00")	379.17	5.42	44.23	83.62	11.17	1.9	52	6.87	11.38	15.65	59.62
8	["2020-11-15 01:45:00","2020-11-15 02:00:00")	398.44	5.66	40.68	76.39	0.02	1.9	52	6.87	11.38	15.65	59.62
9	["2020-11-15 02:00:00","2020-11-15 02:15:00")	398.44	6.2	38.66	71.18	17.84	[null]	56.47	6.96	11.73	15.35	59.9
10	["2020-11-15 02:15:00","2020-11-15 02:30:00")	398.44	8.41	40.7	71.83	14.69	1.76	56.47	6.96	11.73	15.35	59.9
11	["2020-11-15 02:30:00","2020-11-15 02:45:00")	379.17	5.42	44.23	83.62	11.17	1.9	56.47	6.96	11.73	15.35	59.9
12	["2020-11-15 02:45:00","2020-11-15 03:00:00")	545.53	5.58	38.76	74.2	9.72	1.74	56.47	6.96	11.73	15.35	59.9
13	["2020-11-15 03:00:00","2020-11-15 03:15:00")	545.53	7.37	46.77	84.91	15.86	1.87	66.56	7.19	11.42	14.99	60.36
14	["2020-11-15 03:15:00","2020-11-15 03:30:00")	545.53	8.5	49.17	87.74	22.38	1.91	66.56	7.19	11.42	14.99	60.36
15	["2020-11-15 03:30:00","2020-11-15 03:45:00")	525.92	6.75	35.02	63.36	11.89	1.78	66.56	7.19	11.42	14.99	60.36
16	["2020-11-15 03:45:00","2020-11-15 04:00:00")	620.62	7.96	47.45	85.43	19.97	1.94	66.56	7.19	11.42	14.99	60.36
17	["2020-11-15 04:00:00","2020-11-15 04:15:00")	620.62	5.67	44.62	83.81	11						

✓ Successfully run. Total query runtime: 47 msec. 44 rows affected.

FIGURE 4. AQI COMPONENTS ON 15TH NOV 2020 (QUERY 1)

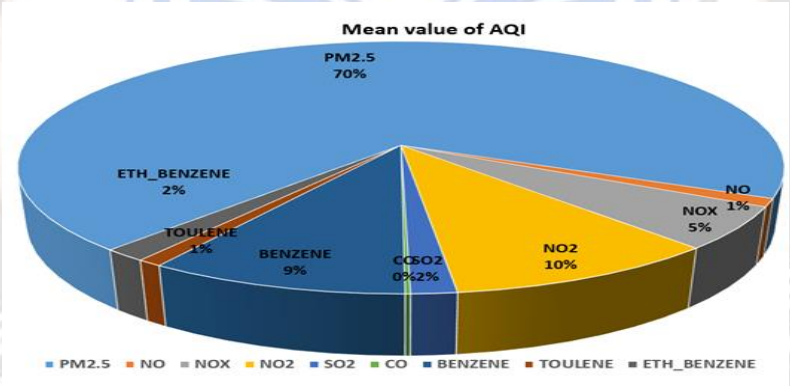


FIGURE 5. THE MEAN PERCENTAGE VALUE OF AQI COMPONENTS ON 15TH NOV 2020

The results of both queries are examined, and it is found that extraction using temporal data modeling is efficient as compared to non-temporal. It took 32% lesser time comparatively. Hence, query performance is noteworthy, and the data extraction is quicker.

Query 2: Find the Air quality index components details when the PM2.5 lies between 100 and 200 µg/m3.

Query Code using SQL Server

```
SELECT * FROM AQI_PER_15_MINS WHERE
AQI_PER_15_MINS.PM25 > 100 INTERSECT SELECT *
FROM AQI_PER_15_MINS WHERE
AQI_PER_15_MINS.PM25 <200;
```

It took 171 milliseconds to complete the data extraction using SQL server with non-temporal database.

Query Code using Postgres with Temporal Data Modeling

```
SELECT AQI.TIMERANGE, AQI."PM25", AQI."NO",
AQI."NOX", AQI."NO2", AQI."SO2", AQI."CO" FROM
(SELECT * FROM PUBLIC."AQI_PER_15_MINS"
WHERE "AQI_PER_15_MINS"."PM25" > 100) AQI
INTERSECT
SELECT AQI_H.TIMERANGE, AQI_H."PM25",
AQI_H."NO", AQI_H."NOX", AQI_H."NO2",
AQI_H."SO2", AQI_H."CO" FROM(SELECT * FROM
PUBLIC."AQI_PER_15_MINS" WHERE
"AQI_PER_15_MINS"."PM25" < 200) AQI_H ORDER BY
TIMERANGE;
```

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 September 2023*

This query is used to retrieve AQI component details throughout timelines during which air pollutant component particle pollution/particulate matter PM2.5 is hazardous and ranges from 100 to 200 µg/m3. The results from the PostgreSQL 10 database are illustrated in figure 6. The results are obtained in only 135 milliseconds. The essential PM2.5 data has been released, and it reveals that 5083 times during the year 2020, the value of PM2.5 repeated, was unhealthy and required the attention of the authorities.

	timerange tsrange	PM25 real	NO real	NOX real	NO2 real	SO2 real	CO real
1	["2020-01-01 02:45:00","2020-01-01 03:00:00")	185.23	[null]	0	[null]	23.99	1.33
2	["2020-01-01 03:00:00","2020-01-01 03:15:00")	168.26	[null]	0	[null]	[null]	[null]
3	["2020-01-01 03:15:00","2020-01-01 03:30:00")	168.26	[null]	0	[null]	16.7	1.29
4	["2020-01-01 03:45:00","2020-01-01 04:00:00")	168.02	[null]	0	[null]	20.56	1.26
5	["2020-01-01 04:00:00","2020-01-01 04:15:00")	167.77	[null]	0	[null]	[null]	1.23
6	["2020-01-01 04:15:00","2020-01-01 04:30:00")	167.77	[null]	0	[null]	16.82	1.23
7	["2020-01-01 04:30:00","2020-01-01 04:45:00")	168.26	[null]	0	[null]	19.6	1.31
8	["2020-01-01 04:45:00","2020-01-01 05:00:00")	172.97	[null]	0	[null]	22.07	1.23
9	["2020-01-01 05:00:00","2020-01-01 05:15:00")	178.41	[null]	0	[null]	[null]	1.28
10	["2020-01-01 05:15:00","2020-01-01 05:30:00")	178.41	[null]	0	[null]	26.23	1.32
11	["2020-01-01 05:30:00","2020-01-01 05:45:00")	167.77	[null]	0	[null]	21.1	1.22
12	["2020-01-01 05:45:00","2020-01-01 06:00:00")	175.42	[null]	0	[null]	29.84	1.31
13	["2020-01-01 06:00:00","2020-01-01 06:15:00")	172.29	[null]	0	[null]	[null]	[null]
14	["2020-01-01 06:15:00","2020-01-01 06:30:00")	172.29	[null]	0	[null]	24.22	1.26
15	["2020-01-01 06:30:00","2020-01-01 06:45:00")	178.41	[null]	0	[null]	27.9	1.28
16	["2020-01-01 06:45:00","2020-01-01 07:00:00")	174.13	[null]	0	[null]	29.61	1.31
17	["2020-01-01 07:00:00","2020-						

✓ Successfully run. Total query runtime: 135 msec. 5083 rows affected.

FIGURE 6. AQI WHEN PM2.5 BETWEEN 100 AND 200 µG/M3 (QUERY 2)

The results of both queries are examined, and it is found that extraction using temporal data modeling is efficient as compared to non-temporal. It took 135 milliseconds only, which is 20% lesser time as compared to conventional data modeling, which takes 171 milliseconds. Hence, query performance is noteworthy with air quality data acquisition through temporal data modeling, and the data extraction is quicker.

Query 3. To generate air quality index details at a particular date and time

Query Code using SQL Server

```
SELECT * FROM AQI_PER_15_MINS A JOIN AQI_PER_HOUR B ON CAST(A.START_TIME AS DATE)= CAST(B.START_TIME AS DATE) AND DATEPART(HOUR,A.START_TIME) =DATEPART(HOUR,B.START_TIME)WHERE A.START_TIME = '2020-11-15 00:00:00.000';
```

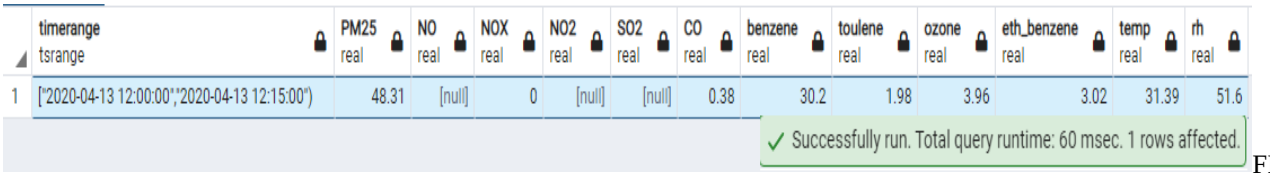
Query executed in 78 milliseconds to complete the extraction of data using conventional data modeling with SQL server.

Query Code using Postgres with Temporal Data Modeling

```
SELECT AQI.TIMERANGE, AQI."PM25", AQI."NO", AQI."NOX", AQI."NO2", AQI."SO2", AQI."CO", AQI_H.BENZENE, AQI_H.TOLUENE, AQI_H.OZONE, AQI_H.ETH_BENZENE, AQI_H.TEMP, AQI_H.RH
FROM (SELECT * FROM PUBLIC."AQI_PER_15_MINS" WHERE PUBLIC."AQI_PER_15_MINS".TIMERANGE @> '2020-04-13 12:00:00'::TIMESTAMP) AQI
INNER JOIN
(SELECT * FROM PUBLIC."AQI_PER_HOUR" WHERE PUBLIC."AQI_PER_HOUR".TIMERANGE @> '2020-04-13 12:00:00'::TIMESTAMP) AQI_H ON AQI.TIMERANGE && AQI_H.TIMERANGE ORDER BY AQI.TIMERANGE;
```

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 September 2023*

This query will return information on the air quality index at a specific day and time. The results from the PostgreSQL 10 database are presented in figure 7, which is optimally acquired in 60 milliseconds out of 35k records. As a result, if decision makers need any attribute at a specific date and time, they can have it in a matter of milliseconds.



	timerange tsrange	PM25 real	NO real	NOX real	NO2 real	SO2 real	CO real	benzene real	toulene real	ozone real	eth_benzene real	temp real	rh real
1	["2020-04-13 12:00:00","2020-04-13 12:15:00")	48.31	[null]	0	[null]	[null]	0.38	30.2	1.98	3.96	3.02	31.39	51.6

Successfully run. Total query runtime: 60 msec. 1 rows affected.

FIGURE 7. AQI DETAILS ON 13TH APRIL 2020 AT 12:00 (QUERY 3)

The results of both queries are examined, and it is observed that extraction using temporal data modeling is efficient as compare to non-temporal. Temporal data modeling extracted data in 60 milliseconds as compared to conventional data modelling which taken only 78 milliseconds and is 23% lesser time comparatively. Hence, query performance is noteworthy with air quality data acquisition through temporal data modeling, and the data extraction is quicker.

Query 4: Show the Trend of Benzene in the month of October 2020 using graphs.

Query Code using SQL Server

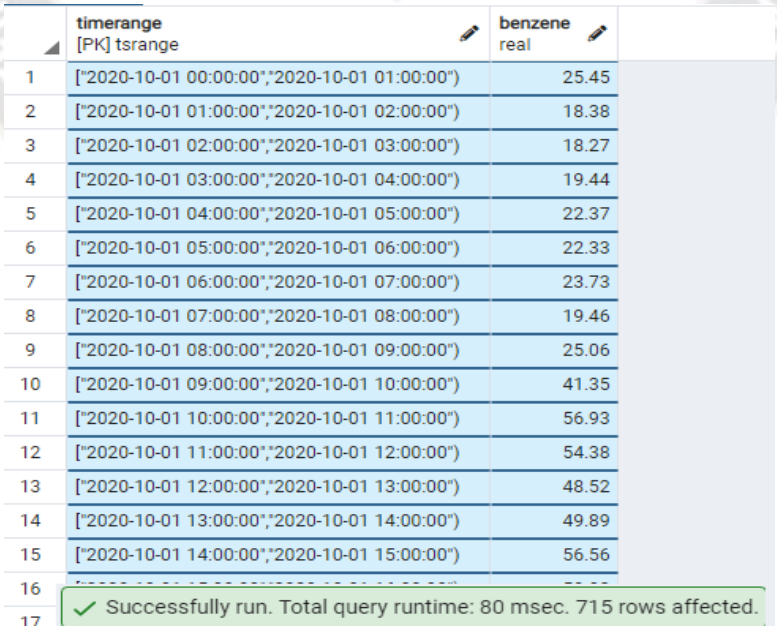
```
SELECT START_TIME, END_TIME, BENZENE FROM AQI_PER_HOUR A WHERE A.START_TIME >= '2020-10-01 00:00:00' AND A.END_TIME <= '2020-10-31 23:59:59' AND BENZENE IS NOT NULL;
```

Conventional relational modeling using SQL server taken 96 milliseconds to complete the extraction of data.

Query Code using Postgres with Temporal Data Modeling

```
SELECT TIMERANGE, BENZENE FROM PUBLIC."AQI_PER_HOUR" WHERE PUBLIC."AQI_PER_HOUR".TIMERANGE && TSRANGE('2020-10-01 00:00:00', '2020-10-31 23:59:59','()') AND BENZENE IS NOT NULL ORDER BY TIMERANGE;
```

This query is run to obtain the Benzene trends for the month of October 2020. The same is represented graphically and can be used for analysis and decision-making. The PostgreSQL 10 database's responses are displayed in figure 8 and graphically in figure 9. In just 80 msecs over the course of a whole month, the best results are produced. In a similar vein, it is possible to quickly acquire trends for any air quality attribute, which may then be utilized for analysis, administration, and decision-making.



	timerange [PK] tsrange	benzene real
1	["2020-10-01 00:00:00","2020-10-01 01:00:00")	25.45
2	["2020-10-01 01:00:00","2020-10-01 02:00:00")	18.38
3	["2020-10-01 02:00:00","2020-10-01 03:00:00")	18.27
4	["2020-10-01 03:00:00","2020-10-01 04:00:00")	19.44
5	["2020-10-01 04:00:00","2020-10-01 05:00:00")	22.37
6	["2020-10-01 05:00:00","2020-10-01 06:00:00")	22.33
7	["2020-10-01 06:00:00","2020-10-01 07:00:00")	23.73
8	["2020-10-01 07:00:00","2020-10-01 08:00:00")	19.46
9	["2020-10-01 08:00:00","2020-10-01 09:00:00")	25.06
10	["2020-10-01 09:00:00","2020-10-01 10:00:00")	41.35
11	["2020-10-01 10:00:00","2020-10-01 11:00:00")	56.93
12	["2020-10-01 11:00:00","2020-10-01 12:00:00")	54.38
13	["2020-10-01 12:00:00","2020-10-01 13:00:00")	48.52
14	["2020-10-01 13:00:00","2020-10-01 14:00:00")	49.89
15	["2020-10-01 14:00:00","2020-10-01 15:00:00")	56.56
16		
17		

Successfully run. Total query runtime: 80 msec. 715 rows affected.

FIGURE. 8 OBSERVATION OF BENZENE IN OCTOBER 2020 (QUERY 4)

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 September 2023*

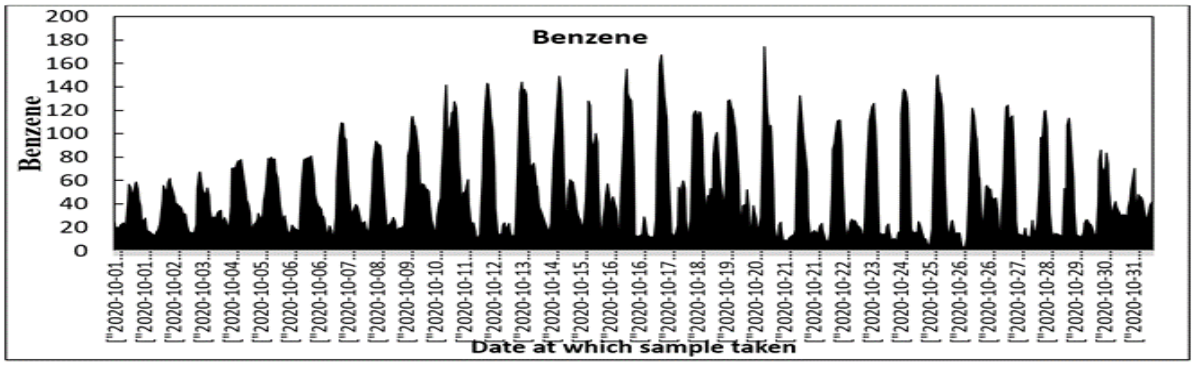


FIGURE 9 TREND OF BENZENE IN OCTOBER 2020

The results of both queries are inspected, and it is found that extraction using temporal data modeling is efficient as compare to non-temporal. It took 17% lesser time comparatively. Hence, query performance is noteworthy with air quality data acquisition through temporal data modeling, and the data extraction is quicker.

Query 5: Show the variations in temperature in the month of June 2020 using graphs.

Query Code using SQL Server

```
SELECT START_TIME , END_TIME, TEMP FROM AQI_PER_HOUR A WHERE A.START_TIME >= '2020-10-01 00:00:00' AND A.END_TIME <= '2020-10-31 23:59:59' AND TEMP IS NOT NULL;
```

Query executed in 89 milliseconds to complete the extraction of data using SQL server with non-temporal database.

Query Code using Postgres with Temporal Data Modeling

```
SELECT TIMERANGE, TEMP FROM PUBLIC."AQI_PER_HOUR" WHERE PUBLIC."AQI_PER_HOUR".TIMERANGE && TSRANGE('2020-06-01 00:00:00', '2020-06-30 23:59:59','[]') AND TEMP IS NOT NULL ORDER BY TIMERANGE;
```

This query is run to collect temperature variations throughout the month of June 2020. June being the hottest month in North India, results are optimally obtained in just 69 msecs for the complete month. A graphical representation of the same can be used for analysis and administration. Figure 10 represents the results obtained and Figure 11 shows the trend of temperature.

	timerange [PK] tsrange	temp real
1	["2020-06-01 00:00:00","2020-06-01 01:00:00")	20.95
2	["2020-06-01 01:00:00","2020-06-01 02:00:00")	21.27
3	["2020-06-01 02:00:00","2020-06-01 03:00:00")	21.36
4	["2020-06-01 03:00:00","2020-06-01 04:00:00")	21.29
5	["2020-06-01 04:00:00","2020-06-01 05:00:00")	21.15
6	["2020-06-01 05:00:00","2020-06-01 06:00:00")	21.13
7	["2020-06-01 06:00:00","2020-06-01 07:00:00")	21.67
8	["2020-06-01 07:00:00","2020-06-01 08:00:00")	23.07
9	["2020-06-01 08:00:00","2020-06-01 09:00:00")	24.62
10	["2020-06-01 09:00:00","2020-06-01 10:00:00")	26.09
11	["2020-06-01 10:00:00","2020-06-01 11:00:00")	27.44
12	["2020-06-01 11:00:00","2020-06-01 12:00:00")	28.9
13	["2020-06-01 12:00:00","2020-06-01 13:00:00")	29.84
14	["2020-06-01 13:00:00","2020-06-01 14:00:00")	30.83
15	["2020-06-01 14:00:00","2020-06-01 15:00:00")	31.47
16	["2020-06-01 15:00:00","2020-06-01 16:00:00")	31.78
17	✓ Successfully run. Total query runtime: 69 msec. 717 rows affected.	

FIGURE 10 RECORDED TEMPERATURE IN MONTH OF JUNE 2020 (QUERY 5)

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 September 2023*

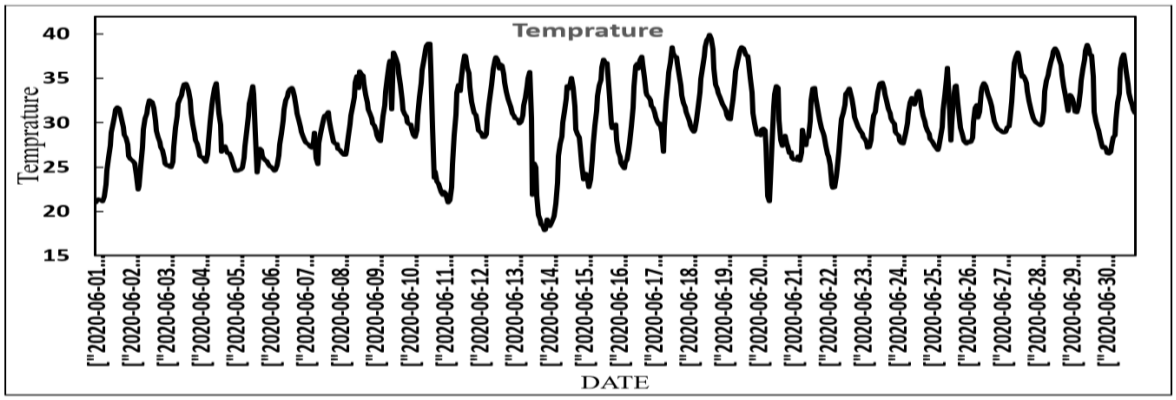


FIGURE 11 TEMP IN MONTH OF JUNE 2020

The results of both queries are studied, and it is found that extraction using temporal data modeling is efficient as compare to non-temporal. It took 22% lesser time comparatively. Hence, query performance is noteworthy with air quality data acquisition through temporal data modeling, and the data extraction is quicker.

V. CONCLUSION

An efficient temporal information system needs a top-notch temporal data model and query language. Air quality indexes (AQI) are employed for forecasting. By modeling future air quality using existing levels, it is possible to make reasonable forecasts for the next few days. If you can access projected AQI, you can evaluate if your planned activity is appropriate given the anticipated air quality on that day. For instance, if you reside in an area where dust storms are common, a projected AQI may be able to inform you that PM levels may be higher on a day when you have outdoor activities scheduled. A variety of institutions, including the central and state pollution control board, and the Ministry of the Environment, analyze data with time variations and make decisions based on this information in order to implement efficient air pollution control measures. The projected AQI helps to reduce pollution, and improve air quality which helps build a better future by significantly lowering the number of sickness and fatalities caused by contaminated atmosphere and sustainable development goals can be achieved. The suggested air quality index model extracts time-varying air quality data using tuple timestamping. Postgres version 10 is used as database management system. The air quality data set collected is utilized to test and determine the outcomes of the proposed model. Valid time is used as the time dimension for capturing data values, together with tuple timestamping. The Postgres "tsrange" datatype is used to record the timestamp. To eliminate data duplication, the dataset is cleaned up and normalized. Using the air quality temporal data model, AQI data at various time points is effectively extracted. It taken only a few milliseconds to extract the AQI data, and the results of the experiment are ready immediately after that. The suggested model is helpful in decision-making and offers decision-makers useful information and the tool they need to enhance air quality. It is helpful in building a better future by significantly lowering the number of sickness and fatalities caused by contaminated atmosphere.

Data Availability

The data used to support this study can be obtained from the corresponding author through formal request.

Conflict of Interest

The authors declare that there is no known conflict of interest.

Funding Statement

The authors declare that this study has not been funded by any funding agency.

REFERENCES

[1] Fuller, R., Landrigan, P. J., Balakrishnan, K., et al., C. (2022). Pollution and health: a progress update. *The Lancet Planetary Health*.

[2] Lelieveld, J., Evans, J. S., Fnais, M., Giannadaki, D., & Pozzer, A. (2015). The contribution of outdoor air pollution sources to premature mortality on a global scale. *Nature*, *525*(7569), 367-371.

[3] Sevilla-Lara, L., Zha S., Yan Z., Goswami V., Feiszli M., Torresani L. “Only Time Can Tell: Discovering Temporal Data for Temporal Modeling” Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2021, pp. 535-544

[4] C. S. Jensen, R. T. Snodgrass, "Temporal data management," IEEE Transactions on Knowledge and Data Engineering,11(1):36-44, 1999. doi:10.1109/69.755613.

[5] Atay, Canan. "A comparison of attribute and tuple time stamped bitemporal relational data models." (2010). Int Conf on Applied Computer Science. 2010: 479-89.

[6] R. Elmasri and S.Navathe, "Fundamentals of Database Systems", Benjamin/Cummings 2012

[7] Kvet, Michal, Karol Matiaško, and Monika Vajsová. "Sensor based transaction temporal database architecture." In 2015 IEEE World Conference on Factory Communication Systems (WFCS), pp. 1-8. IEEE, 2015. DOI: 10.1109/WFCS.2015.7160547

[8] Anselma, L., Terenziani, P., & Snodgrass, R. T. "Valid-time indeterminacy in temporal relational databases: Semantics and representations". IEEE Transactions on Knowledge and Data Engineering, 25(12), pp. 2880-2894, 2013. doi: 10.1109/TKDE.2012.199.

[9] Alromema, Nashwan. "Retrieval optimization technique for tuple timestamp historical relation temporal data model." Journal of Computer Science 8, no. 2 (2012): 243-250.

[10] Petkovic, Dusan. "Temporal Data in Relational Database Systems: A Comparison." New Advances in Information Systems and Technologies. Springer International Publishing, pp. 13-23, 2016. doi: 10.1007/978-3-319-31232-3_2.

[11] Terenziani, Paolo. "Irregular indeterminate repeated facts in temporal relational databases." IEEE Transactions on Knowledge and Data Engineering 28, no. 4 (2015): 1075-1079. doi: 10.1109/TKDE.2015.2509976.

[12] Krause, Josua, Adam Perer, and Harry Stavropoulos. "Supporting iterative cohort construction with visual temporal queries." IEEE Transactions on visualization and computer graphics 22(1), pp. 91-100, 2016. doi: 10.1109/TVCG.2015.2467622.

[13] Kumar, Shailender, and Rahul Rishi. "Retrieval of meteorological data using temporal data modeling." Indian Journal of Science and Technology 9, no. 37 (2016). DOI: 10.17485/ijst/2016/v9i37/99875.

[14] Kumar, Shailender, Rahul Rishi, and Rupender Duggal. "Implementation of temporal functionality in objects with role model (TF-ORM)." In 2016 1st India International Conference on Information Processing (IICIP), pp. 1-5. IEEE, 2016. 10.1109/IICIP.2016.7975368

[15] Bohlen, Michael H., Renato Busatto, and Christian S. Jensen. "Point-versus interval-based temporal data models." In Proceedings 14th international conference on data engineering, pp. 192-200. IEEE, 1998. doi: 10.1109/ICDE.1998.655777.

[16] AL-romema, Nashwan. "Memory storage issues of temporal database applications on relational database management systems." Journal of Computer Science 6, no. 3 (2010).

[17] Murugan, K., and T. Ravichandran. "Intelligent query processing in temporal database using efficient context free grammar." Indian Journal of Science and Technology 5, no. 6 (2012): 1-6.

[18] Ali, Noraida Haji, and Sumazly Sulaiman. "Managing News Archive Using Temporal Data Modeling." Journal of Applied Sciences 12, no. 3 (2012): 284-288.

[19] Wang W., Peng X., Qiao Y., Cheng J. "An empirical study on temporal modeling for online action detection" Complex & Intelligent Systems (2022) 8:1803–1817

[20] "Microsoft Corp", *Microsoft SQL Server*, [online] Available: https://www.microsoft.com/en-us/sql-server