Northeastern University

Assignment 3

Public Housing Inspections Star Schema

Student: Lisa Hoshizuki

NUID: 002767782

College Professional Studies in Analytics, Northeastern University

ALY6030: Data Warehousing & SQL

Professor : Adam Jones

Dec 1, 2024

Table of Contents

Q1. Answer the questions below.

- **How many facts are there in this dataset?**

  Answer: there are two main facts in this dataset: **inspection cost (COST_OF_INSPECTION_IN_DOLLARS)** and **inspection score (INSPECTION_SCORE).**

- **Which facts do you identify?**

  Answer: the inspection cost **(COST_OF_INSPECTION_IN_DOLLARS)** and the inspection score **(INSPECTION_SCORE)** are the two main facts.

- **For the facts that you identify, what type of facts are they?**

  Types of Facts

1. **Additive Facts**: Can be summed across all dimensions. Examples: sales amount, cost, quantity.
2. **Semi-Additive Facts**: Can only be summed across some dimensions but not all. For example, an account balance can be summed across accounts but not over time, since balances are point-in-time values.
3. **Non-Additive Facts**: Cannot be summed at all. For instance, ratios and percentages (like inspection scores or conversion rates) usually don't make sense if added together without specific averaging or weighted approaches.

Answer:

COST_OF_INSPECTION_IN_DOLLARS : Additive Facts

- **Why it's additive**: The COST_OF_INSPECTION_IN_DOLLARS represents the dollar amount spent on each inspection. This cost can be summed across various dimensions such as:

- **Time**: You can sum the inspection costs for each month, quarter, or year to see the total cost of inspections during that period.
- **Location**: You can sum the inspection costs for different cities or states to see the total inspection cost in a particular location.
- **Agency**: You can sum the inspection costs for each public housing agency to see how much they spend on inspections.

Since the COST_OF_INSPECTION_IN_DOLLARS represents a **quantity** (dollars), and summing the costs across different groups or time periods makes sense, it is an **additive fact**.

INSPECTION_SCORE : Semi-Additive Facts

- **Why it's semi-additive**: The INSPECTION_SCORE represents a rating or evaluation of each inspection, often on a scale (e.g., 1 to 100 or 0 to 10). Scores are often intended to assess the **quality** of the inspections.
    - If you were to sum INSPECTION_SCORE across multiple inspections, the result wouldn't represent a meaningful summary of inspection quality. For example, if you summed inspection scores from two buildings (e.g., 80 and 90), you would get a total of 170, which doesn't indicate whether the inspections were good or bad—it just combines the numbers in a way that doesn't make sense.
    - **Average vs. Sum**: Instead of summing scores, you would typically **average** the scores to get a more meaningful result. For example, the average score for two inspections would be (80+90)/2 = 85, which gives you a much more accurate reflection of the overall performance.

Q2. Answer the questions below.

- **How many dimensions are there in this dataset?**

  Answer: Six dimensions can be identified in this dataset.

- **Which dimensions do you identify?**

  Answer:

  Public Housing Agency Name (PUBLIC_HOUSING_AGENCY_NAME)

  Inspected Development Name (INSPECTED_DEVELOPMENT_NAME)

  Inspected Development Address (INSPECTED_DEVELOPMENT_ADDRESS)

  Inspected Development City (INSPECTED_DEVELOPMENT_CITY)

  Inspected Development State (INSPECTED_DEVELOPMENT_STATE)

  Inspection Date (INSPECTION_DATE)

  Why These Are Dimensions:

- PUBLIC_HOUSING_AGENCY_NAME, INSPECTED_DEVELOPMENT_NAME, INSPECTED_DEVELOPMENT_ADDRESS, INSPECTED_DEVELOPMENT_CITY, and INSPECTED_DEVELOPMENT_STATE describe details about the inspection locations and related entities. These descriptive fields provide context for the facts and help categorize them.
- INSPECTION_DATE is a time-based dimension, which allows analysis of inspection data over different time periods. Time dimensions are crucial for grouping and aggregating facts across days, months, or years.

Q3. Answer the question below.

Senior management is interested in viewing the facts identified above, at both the inspection level, as well as a periodic summary of inspection costs for each month. Based on this context, if you were to store these data in a set of fact tables, which type (or types) of fact tables would you use and why?

Answer:

Initial Analysis:

To address senior management's need for both inspection-level details and a periodic summary of inspection costs for each month, the following types of fact tables would be appropriate:

```
-- CREATE TABLE TransactionFactTable
CREATE TABLE `TransactionFactTable` (
    `INSPECTION_ID` INT PRIMARY KEY,
    `PUBLIC_HOUSING_AGENCY_NAME` VARCHAR(100),
    `COST_OF_INSPECTION_IN_DOLLARS` INT,
    `INSPECTED_DEVELOPMENT_NAME` VARCHAR(100),
    `INSPECTED_DEVELOPMENT_ADDRESS` VARCHAR(100),
    `INSPECTED_DEVELOPMENT_CITY` VARCHAR(100),
    `INSPECTED_DEVELOPMENT_STATE` VARCHAR(10),
    `INSPECTION_DATE` DATE,
    `INSPECTION_SCORE` INT
);

-- CREATE TABLE PeriodicSnapshotFactTable
CREATE TABLE `PeriodicSnapshotFactTable` (
    `PUBLIC_HOUSING_AGENCY_NAME` VARCHAR(100) PRIMARY KEY ,
    `SNAPSHOT_DATE` DATE,
    `TOTAL_COST_OF_INSPECTIONS_IN_DOLLARS` INT
);
```

1. **Transaction Fact Table:**
    - **Why:** This fact table stores data at the inspection level, capturing granular information for each inspection. It includes detailed attributes such as INSPECTION_ID,PUBLIC_HOUSING_AGENCY_NAME,INSPECTION_DATE, COST_OF_INSPECTION_IN_DOLLARS, and INSPECTION_SCORE. This table allows for detailed, row-level analysis of each individual inspection, which is important for management to assess the performance and cost of each inspection.

- **Use Case:** Management can use this table to view inspection-specific data such as cost per inspection, the development inspected, and the score received for each inspection. This is useful for identifying trends, inefficiencies, or areas requiring improvement on a per-inspection basis.

2. **Periodic Snapshot Fact Table:**
    - **Why:** This fact table provides a summarized view of inspection costs by month. It aggregates data based on PUBLIC_HOUSING_AGENCY_NAME and SNAPSHOT_DATE (representing the start of each month) and records the total cost of inspections for each month. This table allows for high-level reporting and trend analysis of inspection costs over time.
    - **Use Case:** Senior management can use this table to track monthly inspection costs at the agency level. This is beneficial for budget analysis, forecasting, and understanding overall cost trends.

In conclusion, both types of fact tables—Transaction Fact Table and Periodic Snapshot Fact Table—are required to meet the needs of senior management:

- Transaction Fact Table for detailed, inspection-level data.
- Periodic Snapshot Fact Table for summarized, monthly-level data.

Advanced Analysis:

```
SET GLOBAL local_infile = 1;
-- Insert info into TransactionFactTable
LOAD DATA LOCAL INFILE "/Users/zhangliping/Desktop/NEU/ALY6030/module3/public_housing_inspection_data.csv"
       INTO TABLE `TransactionFactTable`
       FIELDS TERMINATED BY ','
       LINES TERMINATED BY '\n'
       IGNORE 1 LINES
    (`INSPECTION_ID`, `PUBLIC_HOUSING_AGENCY_NAME`, `COST_OF_INSPECTION_IN_DOLLARS`, `INSPECTED_DEVELOPMENT_NAME`,
    `INSPECTED_DEVELOPMENT_ADDRESS`, `INSPECTED_DEVELOPMENT_CITY`, `INSPECTED_DEVELOPMENT_STATE`,
    @INSPECTION_DATE, `INSPECTION_SCORE`)
    SET `INSPECTION_DATE` = STR_TO_DATE(@INSPECTION_DATE, '%m/%d/%Y');
```

505355	Boulder Housing Partners	35740	"Family Sites - AC	"
505356	Housing Authority of the City of	17938	COLORADO SPRINGS	30 S Nevada Ave
505357	Boulder Housing Partners	12992	"Senior Sites - NP	"
505358	Housing Authority of the City of	16290	COLORADO SPRINGS	30 S Nevada Ave

"	4800 Broadway St	Boulder	NULL	5
30 S Nevada Ave	Colorado Springs	CO	2013-05-21	75
"	1133 Portland Pl	Boulder	NULL	5
30 S Nevada Ave	Colorado Springs	CO	2013-06-03	63

After I tried to insert info into table **Transaction Fact Table,** I noticed that column **INSPECTION_DATE** have some null values.

```
SELECT
    SUM(CASE WHEN `INSPECTION_ID` IS NULL THEN 1 ELSE 0 END) AS `INSPECTION_ID_Missing`,
    SUM(CASE WHEN `PUBLIC_HOUSING_AGENCY_NAME` IS NULL THEN 1 ELSE 0 END) AS `PUBLIC_HOUSING_AGENCY_NAME_Missing`,
    SUM(CASE WHEN `COST_OF_INSPECTION_IN_DOLLARS` IS NULL THEN 1 ELSE 0 END) AS `COST_OF_INSPECTION_IN_DOLLARS_Missing`,
    SUM(CASE WHEN `INSPECTED_DEVELOPMENT_NAME` IS NULL THEN 1 ELSE 0 END) AS `INSPECTED_DEVELOPMENT_NAME_Missing`,
    SUM(CASE WHEN `INSPECTED_DEVELOPMENT_ADDRESS` IS NULL THEN 1 ELSE 0 END) AS `INSPECTED_DEVELOPMENT_ADDRESS_Missing`,
    SUM(CASE WHEN `INSPECTED_DEVELOPMENT_CITY` IS NULL THEN 1 ELSE 0 END) AS `INSPECTED_DEVELOPMENT_CITY_Missing`,
    SUM(CASE WHEN `INSPECTED_DEVELOPMENT_STATE` IS NULL THEN 1 ELSE 0 END) AS `INSPECTED_DEVELOPMENT_STATE_Missing`,
    SUM(CASE WHEN `INSPECTION_DATE` IS NULL THEN 1 ELSE 0 END) AS `INSPECTION_DATE_Missing`,
    SUM(CASE WHEN `INSPECTION_SCORE` IS NULL THEN 1 ELSE 0 END) AS `INSPECTION_SCORE_Missing`
FROM `TransactionFactTable`;
```

INSPECTION_ID_Missing	PUBLIC_HOUSING_AGENCY_NAME_Missing	COST_OF_INSPECTION_IN_DOLLARS_Missing	INSPECTED_DEVELOPMENT_NAME_Missing
0	0	0	0

INSPECTED_DEVELOPMENT_ADDRESS_Missing	INSPECTED_DEVELOPMENT_CITY_Missing	INSPECTED_DEVELOPMENT_STATE_Missing	INSPECTION_DATE_Missing	INSPECTION_SCORE_Missing
0	0	0	227	0

Only **INSPECTION_DATE** has 227 missing values.

```
SELECT *,
       (SELECT COUNT(*) FROM `TransactionFactTable` WHERE `INSPECTION_DATE` IS NULL) AS `Null_Date_Count`
FROM `TransactionFactTable`
WHERE `INSPECTION_DATE` IS NULL;
```

INSPECTION_ID	PUBLIC_HOUSING_AGENCY_NAME	COST_OF_INSPE...	INSPECTED_DEVELOPMENT_NAME	INSPECTED_DEVELOPMENT_ADDRESS
506744	Housing Authority of the City of	39956	"PUCKETT HOMES	PAT"
536416	Housing Authority of the City of	39937	"MC HENRY	TYLER
510538	Rowan County Housing Authority	39894	"Running Brook	Gra"
522464	Lucas Metropolitan Housing Autho	39675	"COLLINGWOOD GREEN	"
522093	Housing Authority of Bardstown	39512	"KENNETT	EDELEN
507334	Housing Authority of Cynthiana	38877	"INDIAN HILLS	RIVER"
505987	Paterson Housing Authority	38318	"Barnert	Cotton an"
516136	PUERTO RICO PUBLIC HOUSI...	38284	TURABO HEIGHTS	"Ave. Shufford
516155	PUERTO RICO PUBLIC HOUSI...	38221	ROBERTO CLEMENTE	"Carr. 860
510403	Panama City Housing Authority	38022	"Asbell	Fletcher B"
516119	PUERTO RICO PUBLIC HOUSI...	37902	SAN MARTIN	"Oficina de Administraciion Edi...
511034	PUERTO RICO PUBLIC HOUSI...	37701	"DR JOSE N GANDARA	"

INSPECTED_DEVELOPMENT_ADDRESS	INSPECTED_DEVELOPMEN...	INSPECTED_DEVELOPMENT_ST...	INSPECTION_DATE	INSPECTION_SCORE	Null_...
PAT"	906 Roberts Dr SW	Hartselle	NULL	7	227
TYLER	RAM"	809 W Davi	NULL	0	227
Gra"	2300 Running Brook...	Kannapolis	NULL	10	227
"	800 Division St	Toledo	NULL	4	227
EDELEN	ELM"	513 W Broa	NULL	0	227
RIVER"	148 Federal Dr	Cynthiana	NULL	6	227
Cotton an"	199 Carroll St	Paterson	NULL	7	227
"Ave. Shufford	Edificio 19-I"	Caguas	NULL	1	227
"Carr. 860	Km.2.1	Bo. Marti	NULL	0	227
Fletcher B"	804 E 15th St	Panama Cit	NULL	3	227
"Oficina de Administraciion Edi...	Apt"	Rio Piedra	NULL	11	227
"	Boulevard Miguel A....	Ponce	NULL	9	227

I filtered the data where **INSPECTION_DATE** is null. Then, after checking specific records with **INSPECTION_ID** values of 506744, 536416, and 510538 in the original dataset (public_housing_inspection_data.csv), I found that the inspection dates are not null in those rows. The reason the inspection dates became null in the **TransactionFactTable** is likely due to certain object columns, such as **INSPECTED_DEVELOPMENT_NAME**, containing special characters like commas (",").

INSPECTION_I	PUBLIC_HOUSING_AGENCY_NA	COST_OF_INSPECTION	INSPECTED_DEVELOPMENT_NAME	INSPECTED_DEVELOPMENT_A	INSPECTED_DEVELOPMENT	INSPECTED_DEVELOPMENT	INSPECTION_DA	INSPECTION_SCORI
506744	Housing Authority of the City of	39956	PUCKETT HOMES, PAT	906 Roberts Dr SW	Hartselle	AL	7/29/13	100
536416	Housing Authority of the City of	39937	MC HENRY,TYLER,RAM	809 W Davis St	Hearne	TX	11/21/14	86
510538	Rowan County Housing Authority	39894	Running Brook, Gra	2300 Running Brook Dr	Kannapolis	NC	10/28/13	98

I used python code to clean the dataset as follows.

```python
# Define a function to remove specific characters from strings
def clean_text(text):
    if isinstance(text, str):  # Only process string values
        return text.replace(',', '')
    return text

# Apply the function to columns that need cleaning
columns_to_clean = [
    'PUBLIC_HOUSING_AGENCY_NAME',
    'INSPECTED_DEVELOPMENT_NAME',
    'INSPECTED_DEVELOPMENT_ADDRESS',
    'INSPECTED_DEVELOPMENT_CITY',
    'INSPECTED_DEVELOPMENT_STATE'
]

for col in columns_to_clean:
    df[col] = df[col].apply(clean_text)

# Display the cleaned data
print(df.head())

# Optionally, save the cleaned data back to a CSV file
cleaned_df = '/Users/zhangliping/Desktop/NEU/ALY6030/module3/cleaned_public_housing_inspection_data.csv'
df.to_csv(cleaned_df, index=False)
```

Afterward, I created the **TransactionFactTable**, **PeriodicSnapshotFactTable** and inserted data using **cleaned_public_housing_inspection_data.csv.** I also checked for any null values in the **TransactionFactTable**, and the results show that there are no null values.

```
-- CREATE TABLE TransactionFactTable
CREATE TABLE `TransactionFactTable` (
    `INSPECTION_ID` INT PRIMARY KEY,
    `PUBLIC_HOUSING_AGENCY_NAME` VARCHAR(100),
    `COST_OF_INSPECTION_IN_DOLLARS` INT,
    `INSPECTED_DEVELOPMENT_NAME` VARCHAR(100),
    `INSPECTED_DEVELOPMENT_ADDRESS` VARCHAR(100),
    `INSPECTED_DEVELOPMENT_CITY` VARCHAR(100),
    `INSPECTED_DEVELOPMENT_STATE` VARCHAR(10),
    `INSPECTION_DATE` DATE,
    `INSPECTION_SCORE` INT
);

-- CREATE TABLE PeriodicSnapshotFactTable
CREATE TABLE `PeriodicSnapshotFactTable` (
    `PUBLIC_HOUSING_AGENCY_NAME` VARCHAR(100) PRIMARY KEY ,
    `SNAPSHOT_DATE` DATE,
    `TOTAL_COST_OF_INSPECTIONS_IN_DOLLARS` INT
);
```

```
SET GLOBAL local_infile = 1;
-- Insert info into TransactionFactTable
LOAD DATA LOCAL INFILE
"/Users/zhangliping/Desktop/NEU/ALY6030/module3/cleaned_public_housing_inspection_data.csv"
 INTO TABLE `TransactionFactTable`
 FIELDS TERMINATED BY ','
 LINES TERMINATED BY '\n'
 IGNORE 1 LINES
    (`INSPECTION_ID`, `PUBLIC_HOUSING_AGENCY_NAME`, `COST_OF_INSPECTION_IN_DOLLARS`, `INSPECTED_DEVELOPMENT_NAME` ,
    `INSPECTED_DEVELOPMENT_ADDRESS`, `INSPECTED_DEVELOPMENT_CITY`, `INSPECTED_DEVELOPMENT_STATE`,
    @INSPECTION_DATE, `INSPECTION_SCORE`)
    SET `INSPECTION_DATE` = STR_TO_DATE(@INSPECTION_DATE, '%m/%d/%Y');
```

```
SELECT
    SUM(CASE WHEN `INSPECTION_ID` IS NULL THEN 1 ELSE 0 END) AS `INSPECTION_ID_Missing`,
    SUM(CASE WHEN `PUBLIC_HOUSING_AGENCY_NAME` IS NULL THEN 1 ELSE 0 END) AS `PUBLIC_HOUSING_AGENCY_NAME_Missing`,
    SUM(CASE WHEN `COST_OF_INSPECTION_IN_DOLLARS` IS NULL THEN 1 ELSE 0 END) AS `COST_OF_INSPECTION_IN_DOLLARS_Missing`,
    SUM(CASE WHEN `INSPECTED_DEVELOPMENT_NAME` IS NULL THEN 1 ELSE 0 END) AS `INSPECTED_DEVELOPMENT_NAME_Missing`,
    SUM(CASE WHEN `INSPECTED_DEVELOPMENT_ADDRESS` IS NULL THEN 1 ELSE 0 END) AS `INSPECTED_DEVELOPMENT_ADDRESS_Missing`,
    SUM(CASE WHEN `INSPECTED_DEVELOPMENT_CITY` IS NULL THEN 1 ELSE 0 END) AS `INSPECTED_DEVELOPMENT_CITY_Missing`,
    SUM(CASE WHEN `INSPECTED_DEVELOPMENT_STATE` IS NULL THEN 1 ELSE 0 END) AS `INSPECTED_DEVELOPMENT_STATE_Missing`,
    SUM(CASE WHEN `INSPECTION_DATE` IS NULL THEN 1 ELSE 0 END) AS `INSPECTION_DATE_Missing`,
    SUM(CASE WHEN `INSPECTION_SCORE` IS NULL THEN 1 ELSE 0 END) AS `INSPECTION_SCORE_Missing`
FROM `TransactionFactTable`;
```

INSPECTION_ID_Missing	PUBLIC_HOUSING_AGENCY_NAME_Missing	COST_OF_INSPECTION_IN_DOLLARS_Missing	INSPECTED_DEVELOPMENT_NAME_Missing
0	0	0	0

INSPECTED_DEVELOPMENT_ADDRESS_Mis...	INSPECTED_DEVELOPMENT_CITY_Missing	INSPECTED_DEVELOPMENT_STATE_Missing	INSPECTION_DATE_Missing	INSPECTION_SCORE_Missing
0	0	0	0	0

Part of TransactionFactTable screentshot:

INSPECTION_ID	PUBLIC_HOUSING_AGENCY_NAME	COST_OF_INSPECTION_IN_DOLLARS	INSPECTED_DEVELOPMENT_NAME	INSPECTED_DEVELOPMENT_ADDRESS	INSPECTED_DEVELOPMENT_CITY
500002	MOBILE HOUSING BOARD	28305	FRANK W. BOYKIN TO	1600 Michigan Ave	Mobile
500003	MOBILE HOUSING BOARD	33211	THOMAS JAMES PLACE	1555A Eagle Dr	Mobile
500004	MOBILE HOUSING BOARD	10763	ROGER WILLIAMS HOM	308 Simington Dr	Mobile
500005	MOBILE HOUSING BOARD	34195	GULF VILLAGE	2002 Ball Ave	Prichard
500006	MOBILE HOUSING BOARD	36796	DOWNTOWN RENAISSAN	350 Bloodgood St	Mobile
500007	Statesville Housing Authority	20254	Oaktree Village	110 W Allison St	Statesville
500008	Statesville Housing Authority	35534	PARKWOOD VILLAGE	110 W Allison St	Statesville
500009	Statesville Housing Authority	36063	Raleigh Hills	110 W Allison St	Statesville
500010	MOBILE HOUSING BOARD	28408	R V TAYLOR PLAZA	1517 Plaza Dr	Mobile
500011	Columbus Metropolitan Housi…	27354	ROSEWIND	1400 Brooks Ave	Columbus
500013	Columbus Metropolitan Housi…	16621	SUNSHINE TERRACE	272 S Gift St	Columbus
500014	Columbus Metropolitan Housi…	22488	SAWYER MANOR & TOW	940 Caldwell Pl	Columbus
500015	Columbus Metropolitan Housi…	32034	MARION SQUARE	1316 Marion Rd	Columbus
500017	Alexandria Redevelopment &…	16880	James Bland Phase	600 N Fairfax St	Alexandria
500018	Columbus Metropolitan Housi…	23593	KENMORE SQUARE	1720 Kenmore Rd	Columbus
500019	Columbus Metropolitan Housi…	34129	POST OAK STATION I	1383 Vida Way	Columbus
500020	Columbus Metropolitan Housi…	26358	GLENVIEW ESTATES	4625 Cleveland Ave	Columbus
500021	Columbus Metropolitan Housi…	12408	EASTMOOR SQUARE	59 Alexander Ln	Columbus
500022	Columbus Metropolitan Housi…	15535	THORNWOOD COMMONS	1110 Olmstead Ave	Columbus
500023	Columbus Metropolitan Housi…	11436	TREVITT HEIGHTS II	940 Caldwell Pl	Columbus
500024	Columbus Metropolitan Housi…	22733	THE MEADOWS	4855 Pintail Creek Dr	Canal Winchester
500025	Columbus Metropolitan Housi…	11920	Waggoner Senior Ho	831 Acorn Grove Dr	Blacklick
500026	Columbus Metropolitan Housi…	15131	Worley Terrace II	99 S Central Ave	Columbus

INSPECTED_DEVELOPMENT_CITY	INSPECTED_DEVELOPMENT_STATE	INSPECTION_DATE	INSPECTION_SCORE
Mobile	AL	2012-10-10	90
Mobile	AL	2012-11-06	42
Mobile	AL	2013-04-09	34
Prichard	AL	2012-11-06	76
Mobile	AL	2012-10-11	94
Statesville	NC	2012-10-10	81
Statesville	NC	2012-09-19	87
Statesville	NC	2012-11-06	88
Mobile	AL	2012-10-30	52
Columbus	OH	2012-10-16	74
Columbus	OH	2012-10-17	56
Columbus	OH	2012-11-14	72
Columbus	OH	2012-10-23	87
Alexandria	VA	2013-01-15	90
Columbus	OH	2012-10-23	83
Columbus	OH	2012-10-24	82
Columbus	OH	2012-10-24	89
Columbus	OH	2012-10-18	76
Columbus	OH	2012-10-18	80
Columbus	OH	2012-11-06	61
Canal Winchester	OH	2012-10-16	84
Blacklick	OH	2012-10-16	91
Columbus	OH	2012-10-18	91

Part of PeriodicSnapshotFactTable screentshot:

PUBLIC_HOUSING_AGENCY_NAME	SNAPSHOT_DATE	TOTAL_COST_OF_INSPECTIONS_IN_DOLLARS
Abbotsford Housing Authority	2014-12-01	27217
Abingdon Redevelopment and Housi	2014-05-01	37068
Ada County Housing Authority	2013-07-01	16133
ADAMS METROPOLITAN HOUSI…	2014-01-01	56921
Afton Housing Commission	2014-05-01	25288
Agra Housing Authority	2013-12-01	25313
Ahoskie Housing Authority	2014-09-01	19081
Ainsworth Housing Authority	2014-11-01	35188
Akron Metropolitan Housing Autho	2013-06-01	23866
Alachua County	2013-05-01	39653
ALAMEDA COUNTY HSG AUTH	2013-06-01	29550
Alamo Housing Authority	2014-12-01	38537
Alaska Housing Finance Corporati	2014-05-01	176040
Albany Housing Authority	2013-07-01	32345
Albia Housing Agency	2014-11-01	13269
Albion Housing Authority	2015-01-01	33663
Albion Housing Commission	2013-10-01	20290
Alexander County Housing Authori	2014-04-01	28987
Alexandria Redevelopment & Housi	2012-12-01	24328
Algonac Housing Commission	2014-06-01	29121
ALLEGHENY COUNTY HOUSIN…	2013-08-01	22703
Allen Metropolitan Housing Autho	2014-09-01	11227
Allentown Housing Authority	2013-07-01	106694
Alliance Housing Authority	2013-10-01	10628
Alma Housing Authority	2013-07-01	17154
Altoona Housing Authority	2014-09-01	24813
Amherst Housing Authority	2013-07-01	14373
Anderson Housing Authority	2013-08-01	24757
Ann Arbor Housing Commission	2014-11-01	57406
ANNISTON HA	2013-06-01	38601

Q4. Answer the question below.

Senior Management is also concerned with changes in the names and addresses of the public housing agency names since they tend to get merged with other agencies on a frequent basis. Based on this context, how would handle this slowly changing dimension? Select from types 0,1,2, or 3 from the Kimball reading. Justify your answer.

SCD Type Details

- **Type 0 - Passive:** No action is taken when dimensions change. Some data remains as initially inserted, while others may be overwritten.
- **Type 1 - Overwrite**: No history is kept; the old value is simply overwritten by the new one. Useful for corrections like removing special characters or fixing spelling.
- **Type 2 - New Record:** A new row is added to record changes, preserving full history. Each change creates a new entry, which can be costly for frequently changing dimensions.
- **Type 3 - New Column**: Only the current and previous values are stored in separate columns. This technique captures limited history and is less commonly used.

Answer:

To address the concern of agency name and address changes, **Type 2 Slowly Changing Dimension (SCD)** is the most suitable approach for tracking historical data in cases of frequent changes, such as agency mergers or renaming.

Justification:

1.Track Historical Data: Since agency names and addresses can change due to mergers, acquisitions, or administrative updates, preserving historical records of each agency's name and address at the time of inspections is crucial. Type 2 SCD adds a new row each time there is a change, while maintaining a record of the previous data with timestamps, ensuring historical accuracy.

2. Preserve Data Consistency: By creating a new entry for each change, we can design queries to access agency information as it existed during a specific time period. This guarantees that historical data remains consistent when reviewing past inspection records, even when agency details change.

3. Avoid Overwriting Data: Unlike Type 1 SCD (which overwrites records with the latest data), Type 2 keeps both old and new records intact. This is important for maintaining the integrity of historical data in audit trails, reports, and decision-making, without losing the context of past inspections.

Implementation Code:

```sql
-- ------------------------------------------------------------
-- Q4: How to handle changes in public housing agency names
-- and addresses (slowly changing dimensions).
-- ------------------------------------------------------------

-- Create temporary table staging_periodic_snapshot
CREATE TABLE `staging_periodic_snapshot` (
    `PUBLIC_HOUSING_AGENCY_NAME` VARCHAR(100),
    `SNAPSHOT_DATE` DATE,
    `TOTAL_COST_OF_INSPECTIONS_IN_DOLLARS` INT
);

-- Insert new data into staging_periodic_snapshot
INSERT INTO `staging_periodic_snapshot` (`PUBLIC_HOUSING_AGENCY_NAME`, `SNAPSHOT_DATE`, `TOTAL_COST_OF_INSPECTIONS_IN_DOLLARS`)
SELECT
    `PUBLIC_HOUSING_AGENCY_NAME`,
    DATE_FORMAT(`INSPECTION_DATE`, '%Y-%m-01') AS `SNAPSHOT_DATE`,
    SUM(`COST_OF_INSPECTION_IN_DOLLARS`) AS `TOTAL_COST_OF_INSPECTIONS_IN_DOLLARS`
FROM
    `TransactionFactTable`
GROUP BY
    `PUBLIC_HOUSING_AGENCY_NAME`, `SNAPSHOT_DATE`;


-- Drop and recreate PeriodicSnapshotFactTable with updated schema
DROP TABLE IF EXISTS `PeriodicSnapshotFactTable`;

CREATE TABLE `PeriodicSnapshotFactTable` (
    `PUBLIC_HOUSING_AGENCY_NAME` VARCHAR(100),
    `SNAPSHOT_DATE` DATE,
    `TOTAL_COST_OF_INSPECTIONS_IN_DOLLARS` INT,
    `effective_start_date` DATE NOT NULL,
    `effective_end_date` DATE,
    `is_current` BOOLEAN DEFAULT TRUE,
    PRIMARY KEY (`PUBLIC_HOUSING_AGENCY_NAME`, `SNAPSHOT_DATE`)
);
```

```sql
-- Insert or update records in `PeriodicSnapshotFactTable`
INSERT INTO `PeriodicSnapshotFactTable` (
    `PUBLIC_HOUSING_AGENCY_NAME`,
    `SNAPSHOT_DATE`,
    `TOTAL_COST_OF_INSPECTIONS_IN_DOLLARS`,
    `effective_start_date`,
    `effective_end_date`,
    `is_current`
)
SELECT
    s.`PUBLIC_HOUSING_AGENCY_NAME`,
    s.`SNAPSHOT_DATE`,
    s.`TOTAL_COST_OF_INSPECTIONS_IN_DOLLARS`,
    s.`SNAPSHOT_DATE` AS `effective_start_date`,  -- Set start date to snapshot date
    NULL AS `effective_end_date`,                 -- New records have NULL as end date
    TRUE AS `is_current`
FROM
    `staging_periodic_snapshot` AS s
LEFT JOIN
    `PeriodicSnapshotFactTable` AS p
ON
    s.`PUBLIC_HOUSING_AGENCY_NAME` = p.`PUBLIC_HOUSING_AGENCY_NAME`
    AND s.`SNAPSHOT_DATE` = p.`SNAPSHOT_DATE`
WHERE
    p.`PUBLIC_HOUSING_AGENCY_NAME` IS NULL;       -- Insert only if no matching record exists

-- Mark old records as non-current and set the end date only if there's a newer record
UPDATE `PeriodicSnapshotFactTable` AS p
INNER JOIN `staging_periodic_snapshot` AS s
ON p.`PUBLIC_HOUSING_AGENCY_NAME` = s.`PUBLIC_HOUSING_AGENCY_NAME`
    AND p.`SNAPSHOT_DATE` < s.`SNAPSHOT_DATE`     -- Only for older records
SET
    p.`is_current` = FALSE,
    p.`effective_end_date` = s.`SNAPSHOT_DATE`    -- Set the end date to the new snapshot date
WHERE
    p.`is_current` = TRUE;
```

PUBLIC_HOUSING_AGENCY_NAME	SNAPSHOT_DATE	TOTAL_COST_OF_INSPECTIONS_IN_DO...	effective_start_date	effective_end_date	is_current
Abbotsford Housing Authority	2014-12-01	27217	2014-12-01	NULL	1
Abingdon Redevelopment and Housi	2014-05-01	37068	2014-05-01	NULL	1
Ada County Housing Authority	2013-07-01	16133	2013-07-01	NULL	1
ADAMS METROPOLITAN HOUSI...	2014-01-01	56921	2014-01-01	NULL	1
Afton Housing Commission	2014-05-01	25288	2014-05-01	NULL	1
Agra Housing Authority	2013-12-01	25313	2013-12-01	NULL	1
Ahoskie Housing Authority	2014-09-01	19081	2014-09-01	NULL	1
Ainsworth Housing Authority	2014-11-01	35188	2014-11-01	NULL	1
Akron Metropolitan Housing Autho	2013-06-01	23866	2013-06-01	2013-07-01	0
Akron Metropolitan Housing Autho	2013-07-01	52832	2013-07-01	2013-08-01	0
Akron Metropolitan Housing Autho	2013-08-01	76296	2013-08-01	2014-08-01	0
Akron Metropolitan Housing Autho	2014-07-01	28711	2014-07-01	2014-08-01	0
Akron Metropolitan Housing Autho	2014-08-01	223797	2014-08-01	2014-10-01	0
Akron Metropolitan Housing Autho	2014-09-01	168051	2014-09-01	2014-10-01	0
Akron Metropolitan Housing Autho	2014-10-01	133531	2014-10-01	NULL	1

Here is an explanation of the data provided for the inspection costs of Abbotsford Housing Authority and Akron Metropolitan Housing Authority across different dates, showing the costs and effective periods:

Data Explanation

1. Abbotsford Housing Authority

- Snapshot Date: 2014-12-01
- Total Cost of Inspections: $27,217

- Effective Start Date: 2014-12-01
- Effective End Date: NULL
- Current Status: is_current = 1
- Explanation: This record indicates that, as of 2014-12-01, the inspection cost for Abbotsford Housing Authority was $27,217. The NULL value in effective_end_date suggests that this is the most recent and currently active cost for this agency.

2. Akron Metropolitan Housing Authority

2013-06-01 Record:

- Total Cost of Inspections: $23,866
- Effective Start Date: 2013-06-01
- Effective End Date: 2013-07-01
- Current Status: is_current = 0
- Explanation: This record was effective from 2013-06-01 to 2013-07-01, after which it was replaced by a new inspection cost.

2013-07-01 Record:

- Total Cost of Inspections: $52,832
- Effective Start Date: 2013-07-01
- Effective End Date: 2013-08-01
- Current Status: is_current = 0
- Explanation: This record, starting from 2013-07-01 with a cost of $52,832, was effective until 2013-08-01, when it was updated by the next inspection cost.

In conclusion, the data for Akron Metropolitan Housing Authority illustrates how inspection costs changed over time, with each record reflecting a specific cost for a set effective period. The `is_current` column helps identify whether a record is active or has been replaced by a newer entry.

Q5. Address the scenario below.

Finally, Senior Management is interested in a subset of this data, for only those PHAs that saw an *increase* in the $$ cost of performing an inspection in their jurisdiction. Since none of them are SQL programmers, they've asked your help in performing this analysis by providing a file as your final deliverable with the following columns:

Note that MR stands for "most recent":

- **PHA_NAME,**
- **MR_INSPECTION_DATE,**
- **MR_INSPECTION_COST,**
- **SECOND_MR_INSPECTION_DATE,**
- **SECOND_MR_INSPECTION_COST,**
- **CHANGE_IN_COST**
- **PERCENT_CHANGE_IN_COST**

Management has asked that you perform this function using lead or lag functions in SQL. However, they're concerned that the files when imported into MySQL Workbench may not properly refer to dates using the correct format. If that is the case, they've asked you to investigate how best to convert dates from TEXT to Date format so that the lead/lag functions work as expected.

They've also asked that you filter your dataset to only those PHAs that saw an increase in $$ cost, and that you only list the PHA once with no duplicates to avoid noisy data. Naturally, this would also require you to filter out PHAs that only performed one inspection, so they've asked you to remove those as well.

```
-- ------------------------------------------------------------------------
-- Q5:Filter out the PHA data according to the specified conditions
-- and calculate the change in inspection costs.
-- ------------------------------------------------------------------------
SET SESSION sql_mode=(SELECT REPLACE(@@sql_mode,'ONLY_FULL_GROUP_BY',''));
-- Drop the temporary table if it exists
DROP TEMPORARY TABLE IF EXISTS `TempSnapshotTable`;
CREATE TEMPORARY TABLE TempSnapshotTable AS
SELECT
    PUBLIC_HOUSING_AGENCY_NAME,
    INSPECTION_DATE AS MR_INSPECTION_DATE,
    COST_OF_INSPECTION_IN_DOLLARS AS MR_INSPECTION_COST,
    LEAD(INSPECTION_DATE) OVER (PARTITION BY PUBLIC_HOUSING_AGENCY_NAME
    ORDER BY INSPECTION_DATE DESC) AS SECOND_MR_INSPECTION_DATE,
    LEAD(COST_OF_INSPECTION_IN_DOLLARS) OVER (PARTITION BY PUBLIC_HOUSING_AGENCY_NAME
    ORDER BY INSPECTION_DATE DESC) AS SECOND_MR_INSPECTION_COST,
    ROW_NUMBER() OVER (PARTITION BY PUBLIC_HOUSING_AGENCY_NAME
    ORDER BY INSPECTION_DATE DESC) AS InspectionCount
FROM
    TransactionFactTable;

-- Retrieve data from TempSnapshotTable based on specified conditions
SELECT
    PUBLIC_HOUSING_AGENCY_NAME,
    MR_INSPECTION_DATE,
    MR_INSPECTION_COST,
    SECOND_MR_INSPECTION_DATE,
    SECOND_MR_INSPECTION_COST,
    (SECOND_MR_INSPECTION_COST - MR_INSPECTION_COST) AS CHANGE_IN_COST,
    ((SECOND_MR_INSPECTION_COST - MR_INSPECTION_COST) / MR_INSPECTION_COST) * 100
    AS PERCENT_CHANGE_IN_COST
FROM
    TempSnapshotTable
WHERE
    InspectionCount > 1
    AND (SECOND_MR_INSPECTION_COST - MR_INSPECTION_COST) > 0
GROUP BY
    PUBLIC_HOUSING_AGENCY_NAME;
```

PUBLIC_HOUSING_AGENCY_NAME	MR_INSPECTION_DATE	MR_INSPECTION_COST	SECOND_MR_INSPECTION_DATE	SECOND_MR_INSPECTION_COST	CHANGE_IN_COST	PERCENT_CHANGE_IN_COST
Akron Metropolitan Housing Autho	2014-10-08	15626	2014-10-07	31822	16196	103.6478
Alachua County	2014-05-01	17019	2013-05-16	39653	22634	132.9925
Alaska Housing Finance Corporati	2014-11-13	21366	2014-11-12	23914	2548	11.9255
Albany Housing Authority	2015-01-09	30247	2015-01-08	38537	8290	27.4077
Alexandria Redevelopment & Housi	2014-04-18	14767	2014-04-03	23713	8946	60.5810
ALLEGHENY COUNTY HOUSING…	2015-02-02	36454	2015-01-30	36497	43	0.1180
Allentown Housing Authority	2014-11-14	18989	2014-11-13	27060	8071	42.5036
ALTOONA HOUSING AUTHORITY	2014-11-24	18863	2014-09-15	24813	5950	31.5432
ANNISTON HA	2014-08-21	10785	2013-06-19	38601	27816	257.9138
Area Housing Commission	2013-06-26	22112	2013-06-25	28713	6601	29.8526
Asbury Park Housing Authority	2014-05-21	14987	2013-08-29	39058	24071	160.6125
ASHTABULA METROPOLITAN H…	2014-04-24	13920	2013-07-26	27925	14005	100.6106
Auburn Housing Authority	2013-09-18	30776	2013-07-18	32363	1587	5.1566
Aurora Housing Authority	2014-06-24	12831	2014-04-22	36456	23625	184.1244
Aurora Housing Authority ofthe C	2013-06-11	14570	2013-06-10	21032	6462	44.3514
Austin Housing Authority	2014-06-26	25920	2014-06-23	33220	7300	28.1636
Battle Creek Housing Commission	2015-01-27	15344	2015-01-26	29939	14595	95.1186
Bay City Housing Commission	2014-01-28	16470	2014-01-28	27944	11474	69.6661
Belmont Metropolitan Housing Aut	2014-06-03	29113	2013-07-10	35736	6623	22.7493
Beloit Housing Authority	2013-05-14	14461	2013-05-13	24295	9834	68.0036
Benton Harbor Housing Commission	2014-10-27	18026	2014-01-22	28473	10447	57.9552
Bergen County Housing Authority	2014-05-28	12018	2014-05-22	20481	8463	70.4194

References

ThoughtSpot, T. (2024a, March 26). *Slowly changing dimensions (SCD): 4 types & how to implement*.https://www.thoughtspot.com/data-trends/data-modeling/slowly-changing dimensions-in-data-warehouse