

DAMG 7370 - Designing Advanced Data Architectures for Business Intelligence

FINAL PROJECT

Team Members:

Dharun Karthick Ramaraj

Nitin Sai Varma Indukuri

Nikhil Godalla

Linata Deshmukh

Dataset

Motor Vehicle Collisions - Crashes

The Motor Vehicle Collisions crash table contains details on the crash event. Each row represents a crash event. The Motor Vehicle Collisions data tables contain information from all police reported motor vehicle collisions in NYC. The police report (MV104-AN) is required to be filled out for collisions where someone is injured or killed, or where there is at least \$1000 worth of damage

(https://www.safercar.gov/sites/nhtsa.dot.gov/files/documents/ny_overlay_mv-104an_rev05_2004.pdf). It should be noted that the data is preliminary and subject to change when the MV-104AN forms are amended based on revised crash details. For the most accurate, up to date statistics on traffic fatalities, please refer to the NYPD Motor Vehicle Collisions page (updated weekly) or Vision Zero View (updated monthly).

Due to success of the CompStat program, NYPD began to ask how to apply the CompStat principles to other problems. Other than homicides, the fatal incidents with which police have the most contact with the public are fatal traffic collisions. Therefore, in April 1998, the Department implemented TrafficStat, which uses the CompStat model to work towards improving traffic safety. Police officers complete form MV-104AN for all vehicle collisions. The MV-104AN is a New York State form that has all of the details of a traffic collision. Before implementing Trafficstat, there was no uniform traffic safety data collection procedure for all of the NYPD precincts. Therefore, the Police Department implemented the Traffic Accident Management System (TAMS) in July 1999 in order to collect traffic data in a uniform method across the City. TAMS required the precincts manually enter a few selected MV-104AN fields to collect very basic intersection traffic crash statistics which included the number of accidents, injuries and fatalities. As the years progressed, there grew a need for additional traffic data so that more detailed analyses could be conducted. The Citywide traffic safety initiative, Vision Zero started in the year 2014. Vision Zero further emphasized the need for the collection of more traffic data in order to work towards the Vision Zero goal, which is to eliminate traffic fatalities. Therefore, the Department in March 2016 replaced the TAMS with the new Finest Online Records Management System (FORMS). FORMS enables the police officers to electronically, using a Department cellphone or computer, enter all of the MV-104AN data fields and stores all of the MV-

104AN data fields in the Department's crime data warehouse. Since all of the MV-104AN data fields are now stored for each traffic collision, detailed traffic safety analyses can be conducted as applicable.

Austin Crash Report Data - Crash Level Records

Crash data is obtained from the Texas Department of Transportation (TXDOT) Crash Record Information System (CRIS) database, which is populated by reports submitted by Texas Peace Officers throughout the state, including Austin Police Department (APD), and maintained by TXDOT.

This dataset contains crash-level records for crashes which have occurred in the last ten years. Crash data may take several days or weeks to be initially provided and finalized as it is furnished to the Austin Transportation & Public Works Department, therefore a two-week delay is implemented to help ensure more accurate and complete results.

Please note that the data and information on this website is for informational purposes only. While we seek to provide accurate information, please note that errors may be present and information presented may not be complete.

Traffic Crashes - Crashes

Crash data shows information about each traffic crash on city streets within the City of Chicago limits and under the jurisdiction of Chicago Police Department (CPD). Data are shown as is from the electronic crash reporting system (E-Crash) at CPD, excluding any personally identifiable information. Records are added to the data portal when a crash report is finalized or when amendments are made to an existing report in E-Crash. Data from E-Crash are available for some police districts in 2015, but citywide data are not available until September 2017. About half of all crash reports, mostly minor crashes, are self-reported at the police district by the driver(s) involved and the other half are recorded at the scene by the police officer responding to the crash. Many of the crash parameters, including street condition data, weather condition, and posted speed limits, are recorded by the reporting officer based on best available information at the time, but many of these may disagree with posted information or other assessments on road conditions. If any new or updated information on a crash is received, the reporting officer may amend the crash report at a later time. A traffic crash within the city limits for which CPD is not the responding police agency, typically crashes on interstate highways, freeway ramps, and on local roads along the City

boundary, are excluded from this dataset.

All crashes are recorded as per the format specified in the Traffic Crash Report, SR1050, of the Illinois Department of Transportation. The crash data published on the Chicago data portal mostly follows the data elements in SR1050 form. The current version of the SR1050 instructions manual with detailed information on each data elements is available [here](#).

As per Illinois statute, only crashes with a property damage value of \$1,500 or more or involving bodily injury to any person(s) and that happen on a public roadway and that involve at least one moving vehicle, except bike dooring, are considered reportable crashes. However, CPD records every reported traffic crash event, regardless of the statute of limitations, and hence any formal Chicago crash dataset released by Illinois Department of Transportation may not include all the crashes listed here.

Change 11/21/2023: We have removed the RD_NO (Chicago Police Department report number) for privacy reasons.

Data Profiling

Profiling: ydata_profiling

Austin Crash Report

```
[10]: AustinProfile.to_notebook_iframe()
```

Error displaying widget: model not found
Error displaying widget: model not found
Error displaying widget: model not found

Austin Crash Report

Overview Variables Interactions Missing values Sample

Overview Alerts 53 Reproduction

Dataset statistics		Variable types	
Number of variables	54	Numeric	17
Number of observations	147750	Boolean	11
Missing cells	1725084	DateTime	2
Missing cells (%)	21.6%	Text	7
Duplicate rows	0	Unsupported	2
Duplicate rows (%)	0.0%	Categorical	15
Total size in memory	60.9 MiB		
Average record size in memory	432.0 B		

Austin Crash Report

Overview Variables Interactions Missing values Sample

Overview Alerts 53 Reproduction

Alerts

pedestrian_f1 has constant value ""	Constant
motor_vehicle_f1 has constant value ""	Constant
motorcycle_f1 has constant value ""	Constant
bicycle_f1 has constant value ""	Constant
other_f1 has constant value ""	Constant
other_death_count has constant value ""	Constant
private_dr_f1 has constant value ""	Constant
micromobility_f1 has constant value ""	Constant

Chicago Crash Report

Chicago Crash Report

Overview Variables Interactions Missing values Sample

Overview

Overview		Alerts 35	Reproduction
Dataset statistics			Variable types
Number of variables	48	Text	3
Number of observations	817723	Boolean	9
Missing cells	8268003	DateTime	2
Missing cells (%)	21.1%	Numeric	15
Duplicate rows	0	Categorical	19
Duplicate rows (%)	0.0%		
Total size in memory	299.5 MiB		
Average record size in memory	384.0 B		

Chicago Crash Report

Overview Variables Interactions Missing values Sample

Alerts

INJURIES_UNKNOWN has constant value ""	Constant
TRAFFIC_CONTROL_DEVICE is highly imbalanced (61.7%)	Imbalance
DEVICE_CONDITION is highly imbalanced (54.4%)	Imbalance
WEATHER_CONDITION is highly imbalanced (66.2%)	Imbalance
ALIGNMENT is highly imbalanced (92.0%)	Imbalance
ROADWAY_SURFACE_COND is highly imbalanced (55.3%)	Imbalance
ROAD_DEFECT is highly imbalanced (69.9%)	Imbalance
INTERSECTION RELATED_I is highly imbalanced (72.4%)	Imbalance
NOT_RIGHT_OF_WAY_I is highly imbalanced (55.8%)	Imbalance
HIT_AND_RUN_I is highly imbalanced (74.4%)	Imbalance
SEC_CONTRIBUTORY_CAUSE is highly imbalanced (54.2%)	Imbalance
MOST_SEVERE_INJURY is highly imbalanced (66.7%)	Imbalance
INJURIES_FATAL is highly imbalanced (99.4%)	Imbalance
CRASH_DATE EST T has 758504 (92.5%) missing values	Missing

NY Crash Report

NY Crash Report

Overview Variables Interactions Missing values Sample

Overview

Overview		Alerts 31	Reproduction
Dataset statistics			Variable types
Number of variables	29	Date Time	2
Number of observations	2075427	Categorical	6
Missing cells	17761579	Unsupported	1
Missing cells (%)	29.5%	Numeric	8
Duplicate rows	0	Text	12
Duplicate rows (%)	0.0%		
Total size in memory	459.2 MiB		
Average record size in memory	232.0 B		

Overview Alerts 31 Reproduction

Alerts

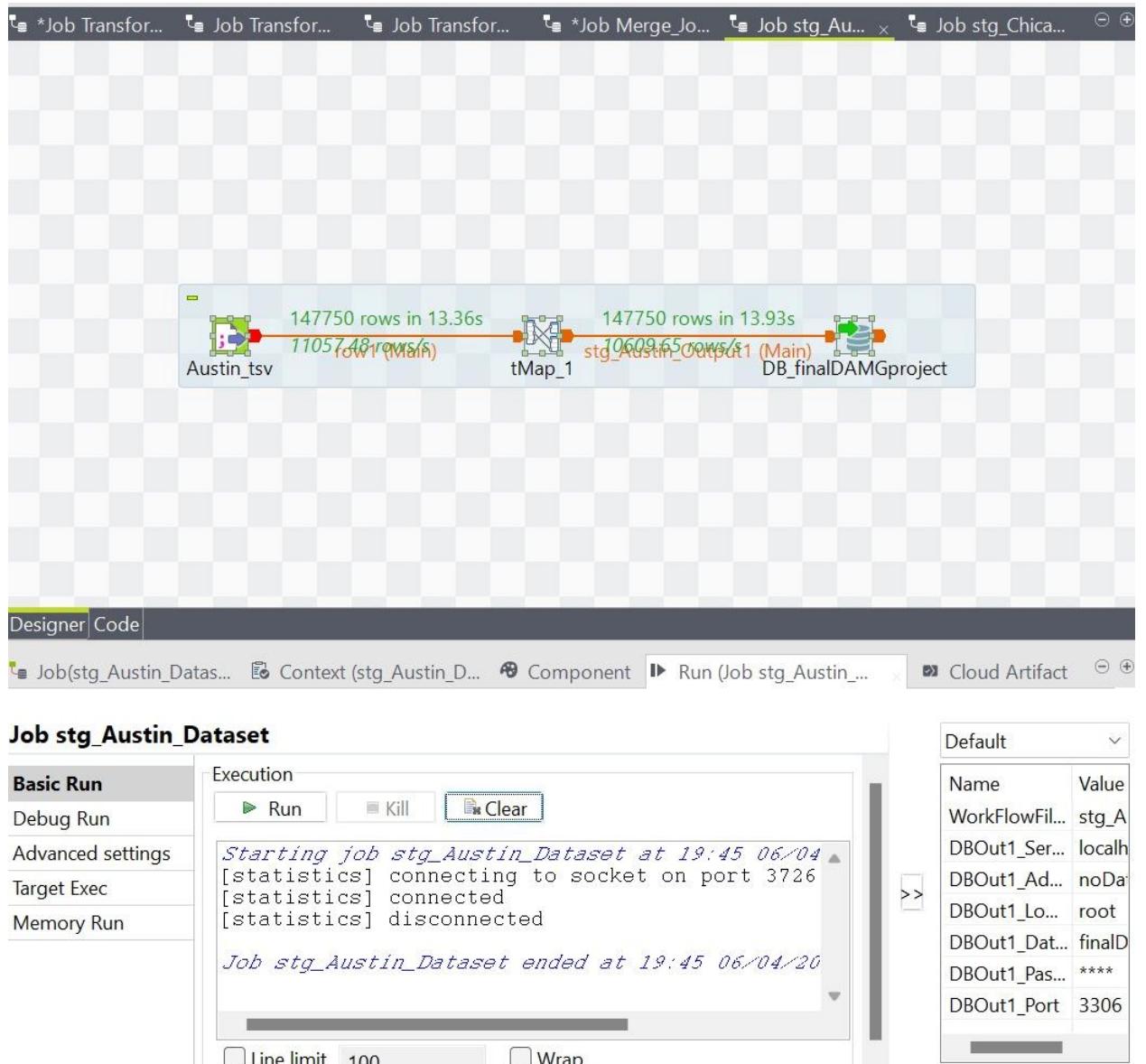
NUMBER OF PEDESTRIANS KILLED	is highly imbalanced (99.6%)	Imbalance
NUMBER OF CYCLIST INJURED	is highly imbalanced (92.3%)	Imbalance
NUMBER OF CYCLIST KILLED	is highly imbalanced (99.9%)	Imbalance
CONTRIBUTING FACTOR VEHICLE 4	is highly imbalanced (90.8%)	Imbalance
CONTRIBUTING FACTOR VEHICLE 5	is highly imbalanced (89.9%)	Imbalance
BOROUGH	has 645746 (31.1%) missing values	Missing
ZIP CODE	has 645996 (31.1%) missing values	Missing
LATITUDE	has 233626 (11.3%) missing values	Missing
LONGITUDE	has 233626 (11.3%) missing values	Missing
LOCATION	has 233626 (11.3%) missing values	Missing
ON STREET NAME	has 440569 (21.2%) missing values	Missing
CROSS STREET NAME	has 784436 (37.8%) missing values	Missing
OFF STREET NAME	has 1727231 (83.2%) missing values	Missing

Staging

Austin

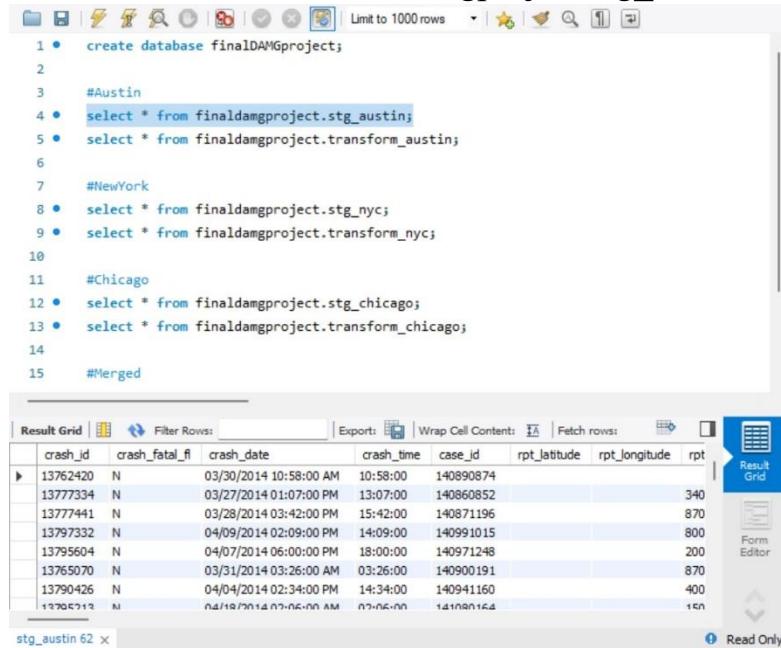
Stage Table: Stg_Austin_Dataset

Purpose: Staging Austin Dataset from tsv file



MYSQL Command

Table - select * from finaldamgproject.stg_austin;



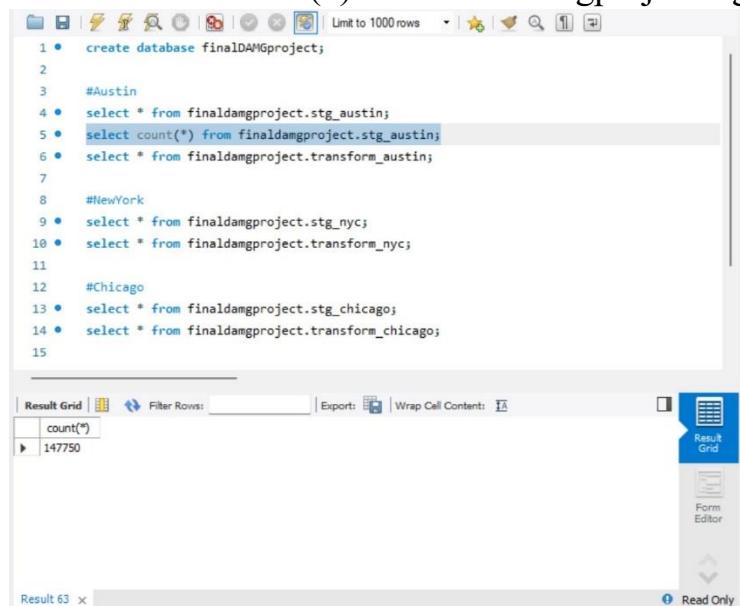
The screenshot shows the MySQL Workbench interface with a query editor and a results grid. The query editor contains the following code:

```
1 • create database finalDAMGproject;
2
3     #Austin
4 • select * from finaldamgproject.stg_austin;
5 • select * from finaldamgproject.transform_austin;
6
7     #NewYork
8 • select * from finaldamgproject.stg_nyc;
9 • select * from finaldamgproject.transform_nyc;
10
11    #Chicago
12 • select * from finaldamgproject.stg_chicago;
13 • select * from finaldamgproject.transform_chicago;
14
15    #Merged
```

The results grid displays the data from the stg_austin table, showing 14 rows of crash information. The columns are: crash_id, crash_fatal_f, crash_date, crash_time, case_id, rpt_latitude, rpt_longitude, and rpt.

crash_id	crash_fatal_f	crash_date	crash_time	case_id	rpt_latitude	rpt_longitude	rpt
13762420	N	03/30/2014 10:58:00 AM	10:58:00	140890874			
13777334	N	03/27/2014 01:07:00 PM	13:07:00	140860852			340
13777441	N	03/28/2014 03:42:00 PM	15:42:00	140871196			870
13797332	N	04/09/2014 02:09:00 PM	14:09:00	140991015			800
13795604	N	04/07/2014 06:00:00 PM	18:00:00	140971248			200
13765070	N	03/31/2014 03:26:00 AM	03:26:00	140900191			870
13790426	N	04/04/2014 02:34:00 PM	14:34:00	140941160			400
13790427	N	04/18/2014 07:06:00 AM	07:06:00	14100164			160

Count - select count(*) from finaldamgproject.stg_austin



The screenshot shows the MySQL Workbench interface with a query editor and a results grid. The query editor contains the following code:

```
1 • create database finalDAMGproject;
2
3     #Austin
4 • select * from finaldamgproject.stg_austin;
5 • select count(*) from finaldamgproject.stg_austin;
6 • select * from finaldamgproject.transform_austin;
7
8     #NewYork
9 • select * from finaldamgproject.stg_nyc;
10 • select * from finaldamgproject.transform_nyc;
11
12    #Chicago
13 • select * from finaldamgproject.stg_chicago;
14 • select * from finaldamgproject.transform_chicago;
15
```

The results grid displays the result of the count query, which is 147750.

count(*)
147750

DDL commands :

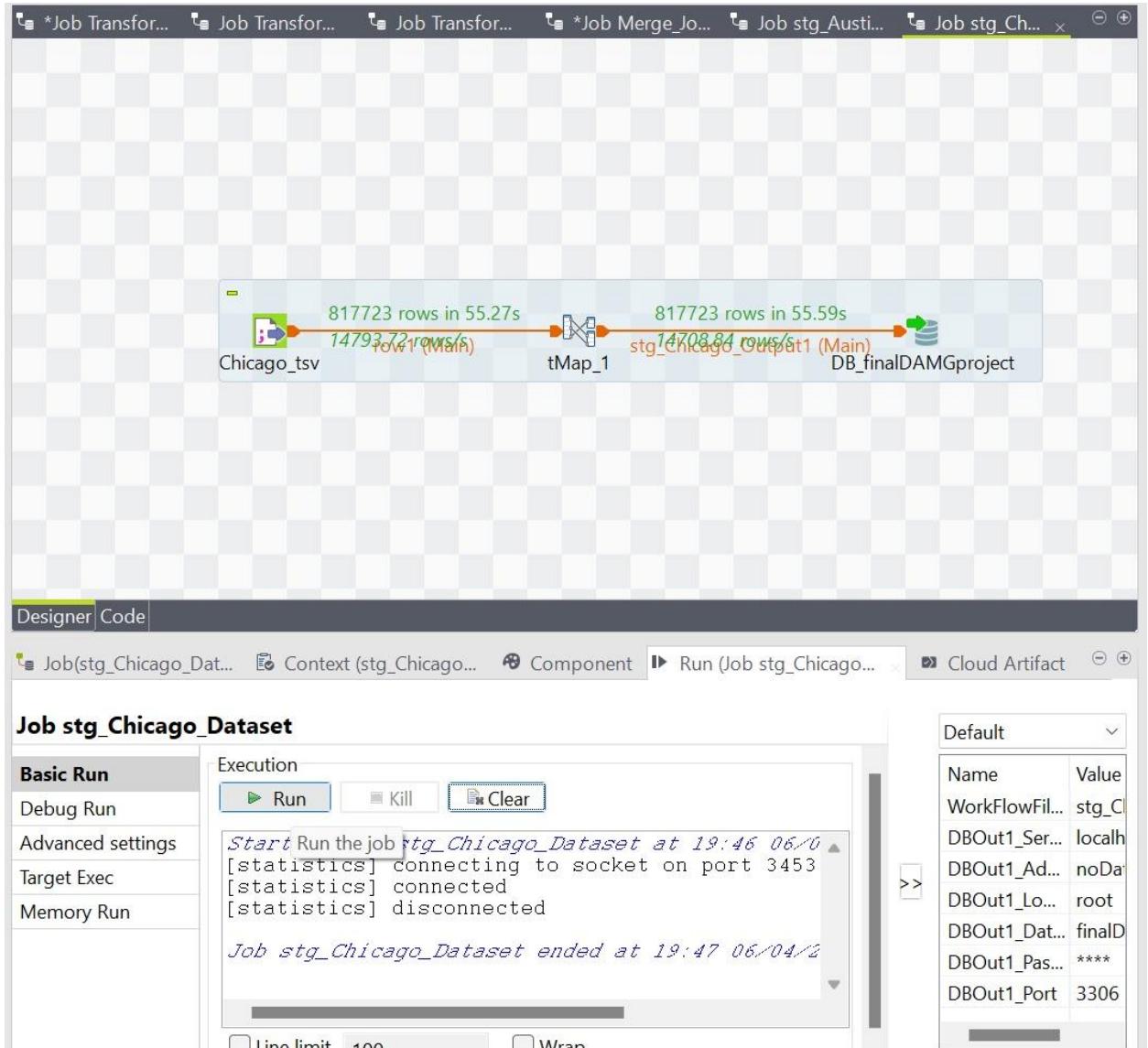
```
CREATE TABLE stg_austin (
    crash_id varchar(100) DEFAULT NULL,
    crash_fatal_flg varchar(100) DEFAULT NULL,
    crash_date varchar(100) DEFAULT NULL,
    crash_time varchar(100) DEFAULT NULL,
    case_id varchar(100) DEFAULT NULL,
    rpt_latitude varchar(100) DEFAULT NULL,
    rpt_longitude varchar(100) DEFAULT NULL,
    rpt_block_num varchar(100) DEFAULT NULL,
    rpt_street_pfx varchar(100) DEFAULT NULL,
    rpt_street_name varchar(100) DEFAULT NULL,
    rpt_street_sfx varchar(100) DEFAULT NULL,
    crash_speed_limit varchar(100) DEFAULT NULL,
    road_constr_zone_flg varchar(100) DEFAULT NULL,
    latitude varchar(100) DEFAULT NULL,
    longitude varchar(100) DEFAULT NULL,
    street_name varchar(100) DEFAULT NULL,
    street_nbr varchar(100) DEFAULT NULL,
    street_name_2 varchar(100) DEFAULT NULL,
    street_nbr_2 varchar(100) DEFAULT NULL,
    crash_sev_id int DEFAULT NULL,
    sus_serious_injry_cnt int DEFAULT NULL,
    nonincap_injry_cnt varchar(100) DEFAULT NULL,
    poss_injry_cnt varchar(100) DEFAULT NULL,
    non_injry_cnt varchar(100) DEFAULT NULL,
    unkn_injry_cnt varchar(100) DEFAULT NULL,
    tot_injry_cnt varchar(100) DEFAULT NULL,
    death_cnt int DEFAULT NULL,
    contrib_factr_p1_id varchar(100) DEFAULT NULL,
    contrib_factr_p2_id varchar(100) DEFAULT NULL,
    units_involved varchar(500) DEFAULT NULL,
    atd_mode_category_metadata varchar(6000) DEFAULT NULL,
    pedestrian_flg varchar(100) DEFAULT NULL,
    motor_vehicle_flg varchar(100) DEFAULT NULL,
```

```
motorcycle_fl varchar(100) DEFAULT NULL,  
bicycle_fl varchar(100) DEFAULT NULL,  
other_fl varchar(100) DEFAULT NULL,  
point varchar(45) DEFAULT NULL,  
apd_confirmed_fatality varchar(100) DEFAULT NULL,  
apd_confirmed_death_count int DEFAULT NULL,  
motor_vehicle_death_count int DEFAULT NULL,  
motor_vehicle_serious_injury_count int DEFAULT NULL,  
bicycle_death_count int DEFAULT NULL,  
bicycle_serious_injury_count int DEFAULT NULL,  
pedestrian_death_count int DEFAULT NULL,  
pedestrian_serious_injury_count int DEFAULT NULL,  
motorcycle_death_count int DEFAULT NULL,  
motorcycle_serious_injury_count int DEFAULT NULL,  
other_death_count int DEFAULT NULL,  
other_serious_injury_count int DEFAULT NULL,  
onsys_fl varchar(100) DEFAULT NULL,  
private_dr_fl varchar(100) DEFAULT NULL,  
micromobility_serious_injury_count int DEFAULT NULL,  
micromobility_death_count int DEFAULT NULL,  
micromobility_fl varchar(100) DEFAULT NULL,  
DI_CreatedDate datetime DEFAULT NULL,  
DI_WorkFlowFileName varchar(100) DEFAULT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4  
COLLATE=utf8mb4_0900_ai_ci
```

Chicago

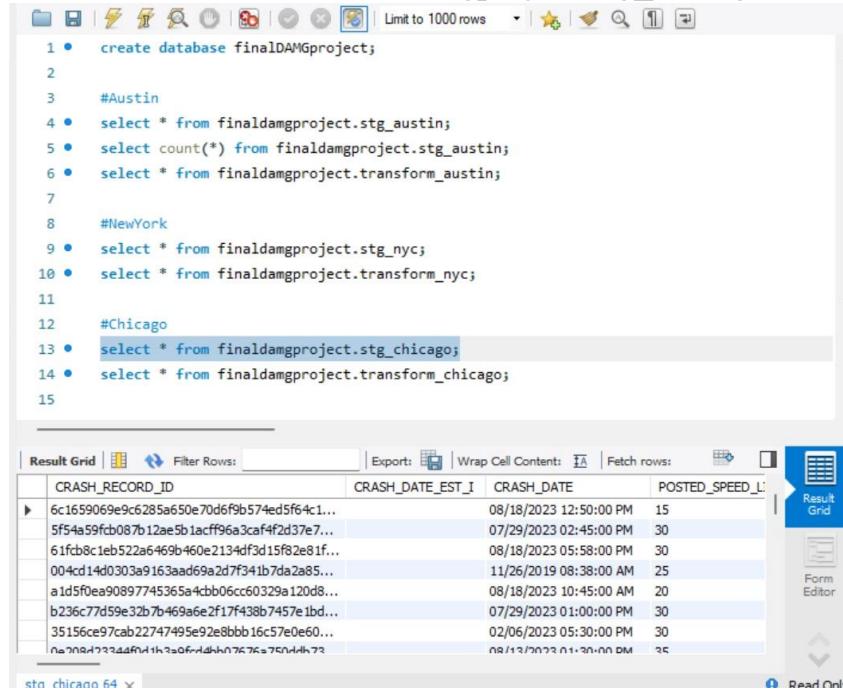
Stage Table: Stg_Chicago_Dataset

Purpose: Staging Chicago Dataset from tsv file



MYSQL Command

Table - select * from finaldamgproject.stg_chicago;



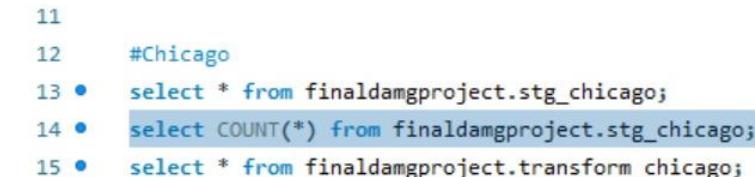
The screenshot shows the MySQL Workbench interface with a query editor and a result grid. The query editor contains the following code:

```
1 •  create database finalDAMGproject;
2
3     #Austin
4 •  select * from finaldamgproject.stg_austin;
5 •  select count(*) from finaldamgproject.stg_austin;
6 •  select * from finaldamgproject.transform_austin;
7
8     #NewYork
9 •  select * from finaldamgproject.stg_nyc;
10 • select * from finaldamgproject.transform_nyc;
11
12     #Chicago
13 •  select * from finaldamgproject.stg_chicago;
14 • select * from finaldamgproject.transform_chicago;
15
```

The line `13 • select * from finaldamgproject.stg_chicago;` is highlighted in blue, indicating it is the current query being executed. The result grid below shows the data from the `stg_chicago` table:

CRASH_RECORD_ID	CRASH_DATE_EST_I	CRASH_DATE	POSTED_SPEED_L
6c1659069e9c6285a650e70d6f9b574ed5f64c1...	08/18/2023 12:50:00 PM	15	
5f54a59fc087b12ae5b1acf96a3caf4f2d37e7...	07/29/2023 02:45:00 PM	30	
61fc8c1eb522a6469b460e2134fd3d15f82e81f...	08/18/2023 05:58:00 PM	30	
004cd14d0303a9163aad69a2d7f341b7da2a85...	11/26/2019 08:38:00 AM	25	
a1d5f0ea90897745365a4cb06cc60329a120d8...	08/18/2023 10:45:00 AM	20	
b236c77d59e32b7b469a6e2f17f438b7457e1bd...	07/29/2023 01:00:00 PM	30	
35156ce97cab22747495e92e8bbb16c57e0e60...	02/06/2023 05:30:00 PM	30	
na7n8d17734d8f411h3a0frldhh7k=75ndhd73	08/13/2023 01:30:00 PM	35	

Count - select COUNT(*) from finaldamgproject.stg_chicago;



The screenshot shows the MySQL Workbench interface with a query editor and a result grid. The query editor contains the following code:

```
11
12     #Chicago
13 •  select * from finaldamgproject.stg_chicago;
14 •  select COUNT(*) from finaldamgproject.stg_chicago;
15 •  select * from finaldamgproject.transform_chicago;
```

The line `14 • select COUNT(*) from finaldamgproject.stg_chicago;` is highlighted in blue, indicating it is the current query being executed. The result grid shows the output:

COUNT(*)
817723

DDL commands

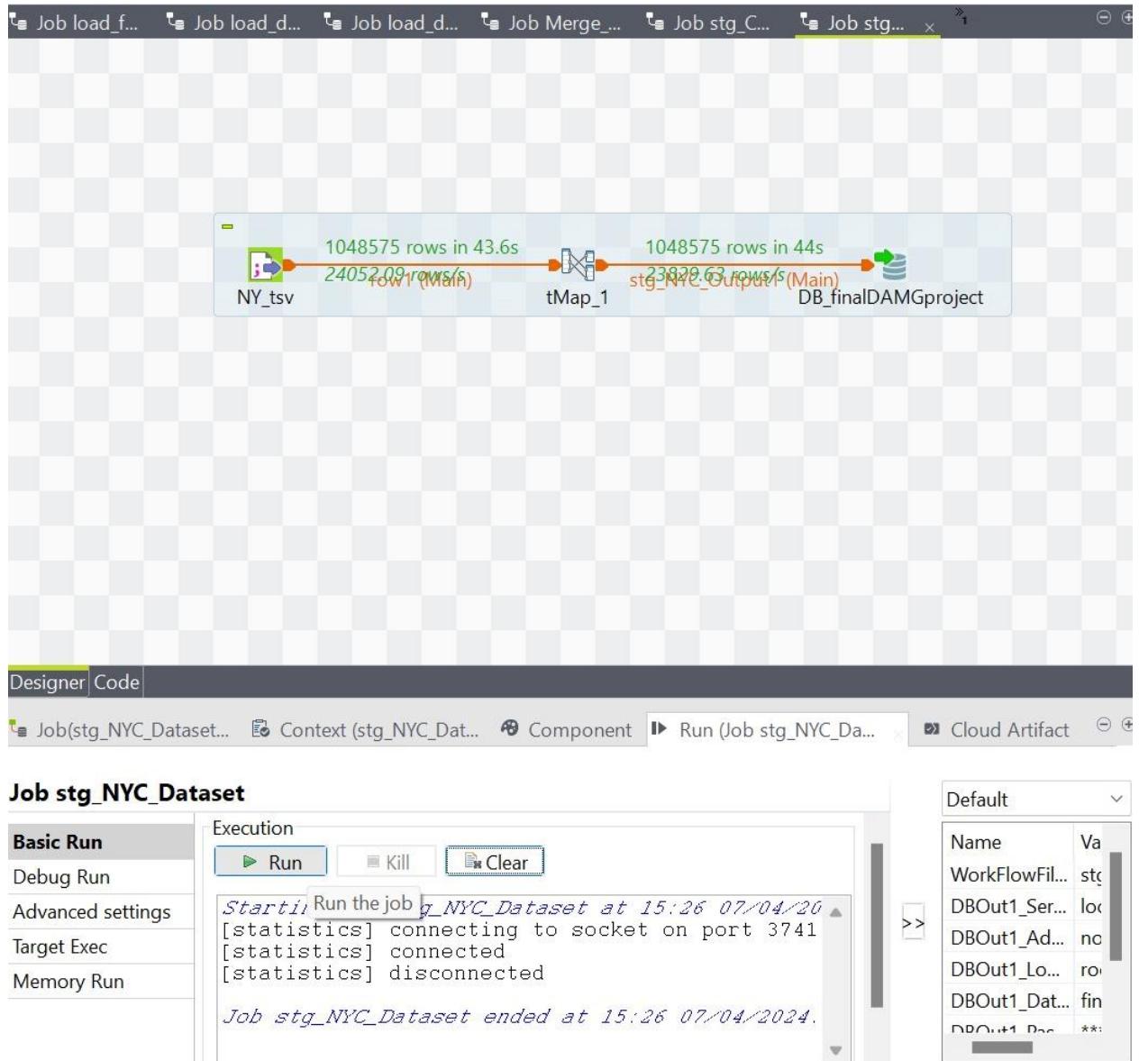
```
CREATE TABLE stg_chicago (
    CRASH_RECORD_ID varchar(500) DEFAULT NULL,
    CRASH_DATE_EST_I varchar(100) DEFAULT NULL,
    CRASH_DATE varchar(100) DEFAULT NULL,
    POSTED_SPEED_LIMIT int DEFAULT NULL,
    TRAFFIC_CONTROL_DEVICE varchar(100) DEFAULT NULL,
    DEVICE_CONDITION varchar(100) DEFAULT NULL,
    WEATHER_CONDITION varchar(100) DEFAULT NULL,
    LIGHTING_CONDITION varchar(100) DEFAULT NULL,
    FIRST_CRASH_TYPE varchar(100) DEFAULT NULL,
    TRAFFICWAY_TYPE varchar(100) DEFAULT NULL,
    LANE_CNT varchar(100) DEFAULT NULL,
    ALIGNMENT varchar(100) DEFAULT NULL,
    ROADWAY_SURFACE_COND varchar(100) DEFAULT NULL,
    ROAD_DEFECT varchar(100) DEFAULT NULL,
    REPORT_TYPE varchar(100) DEFAULT NULL,
    CRASH_TYPE varchar(100) DEFAULT NULL,
    INTERSECTION RELATED_I varchar(100) DEFAULT NULL,
    NOT_RIGHT_OF_WAY_I varchar(100) DEFAULT NULL,
    HIT_AND_RUN_I varchar(100) DEFAULT NULL,
    DAMAGE varchar(100) DEFAULT NULL,
    DATE_POLICE_NOTIFIED varchar(100) DEFAULT NULL,
    PRIM_CONTRIBUTORY_CAUSE varchar(100) DEFAULT NULL,
    SEC_CONTRIBUTORY_CAUSE varchar(100) DEFAULT NULL,
    STREET_NO int DEFAULT NULL,
    STREET_DIRECTION varchar(100) DEFAULT NULL,
    STREET_NAME varchar(100) DEFAULT NULL,
    BEAT_OF_OCCURRENCE int DEFAULT NULL,
    PHOTOS_TAKEN_I varchar(100) DEFAULT NULL,
    STATEMENTS_TAKEN_I varchar(100) DEFAULT NULL,
    DOORING_I varchar(100) DEFAULT NULL,
    WORK_ZONE_I varchar(100) DEFAULT NULL,
    WORK_ZONE_TYPE varchar(100) DEFAULT NULL,
    WORKERS_PRESENT_I varchar(100) DEFAULT NULL,
    NUM_UNITS int DEFAULT NULL,
```

```
MOST_SEVERE_INJURY varchar(100) DEFAULT NULL,  
INJURIES_TOTAL int DEFAULT NULL,  
INJURIES_FATAL int DEFAULT NULL,  
INJURIES_INCAPACITATING int DEFAULT NULL,  
INJURIES_NON_INCAPACITATING int DEFAULT NULL,  
INJURIES_REPORTED_NOT_EVIDENT int DEFAULT NULL,  
INJURIES_NO_INDICATION int DEFAULT NULL,  
INJURIES_UNKNOWN int DEFAULT NULL,  
CRASH_HOUR int DEFAULT NULL,  
CRASH_DAY_OF_WEEK int DEFAULT NULL,  
CRASH_MONTH int DEFAULT NULL,  
LATITUDE varchar(100) DEFAULT NULL,  
LONGITUDE varchar(100) DEFAULT NULL,  
LOCATION varchar(100) DEFAULT NULL,  
DI_CreatedDate datetime DEFAULT NULL,  
DI_WorkFlowFileName varchar(100) DEFAULT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4  
COLLATE=utf8mb4_0900_ai_ci
```

NewYork

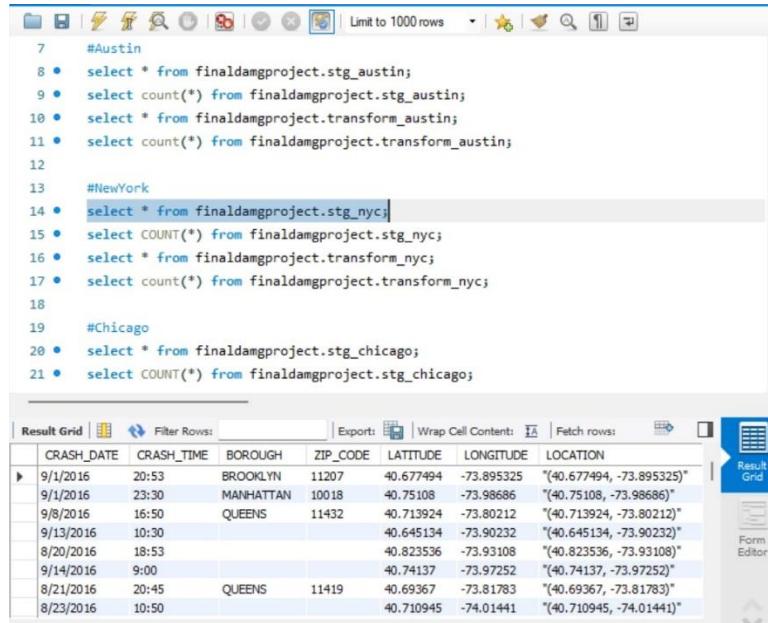
Stage Table: Stg_NYC_Dataset

Purpose: Staging New York Dataset from tsv file



MYSQL Command

Table - select * from finaldamgproject.stg_nyc;



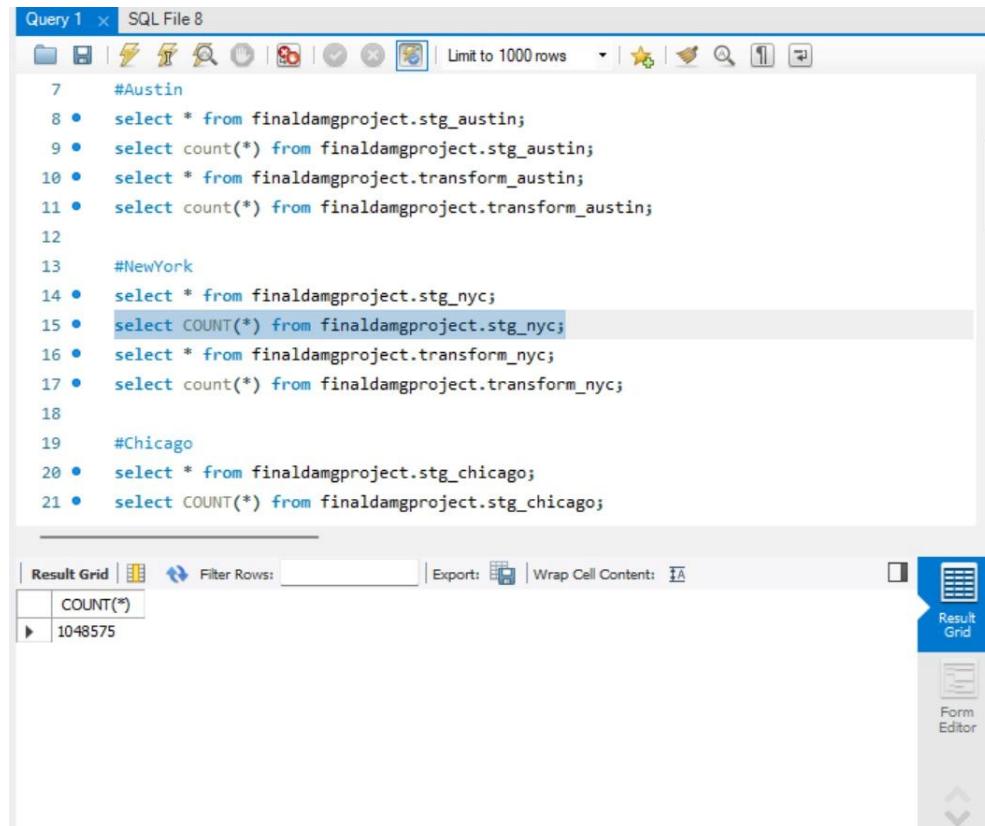
The screenshot shows the MySQL Workbench interface with a query editor and a result grid. The query editor contains the following code:

```
7 #Austin
8 • select * from finaldamgproject.stg_austin;
9 • select count(*) from finaldamgproject.stg_austin;
10 • select * from finaldamgproject.transform_austin;
11 • select count(*) from finaldamgproject.transform_austin;
12
13 #NewYork
14 • select * from finaldamgproject.stg_nyc;
15 • select COUNT(*) from finaldamgproject.stg_nyc;
16 • select * from finaldamgproject.transform_nyc;
17 • select count(*) from finaldamgproject.transform_nyc;
18
19 #Chicago
20 • select * from finaldamgproject.stg_chicago;
21 • select COUNT(*) from finaldamgproject.stg_chicago;
```

The result grid displays the following data:

	CRASH_DATE	CRASH_TIME	BOROUGH	ZIP_CODE	LATITUDE	LONGITUDE	LOCATION
▶	9/1/2016	20:53	BROOKLYN	11207	40.677494	-73.895325	"(40.677494, -73.895325)"
	9/1/2016	23:30	MANHATTAN	10018	40.75108	-73.98686	"(40.75108, -73.98686)"
	9/8/2016	16:50	QUEENS	11432	40.713924	-73.80212	"(40.713924, -73.80212)"
	9/13/2016	10:30			40.645134	-73.90232	"(40.645134, -73.90232)"
	8/20/2016	18:53			40.823536	-73.93108	"(40.823536, -73.93108)"
	9/14/2016	9:00			40.74137	-73.97252	"(40.74137, -73.97252)"
	8/21/2016	20:45	QUEENS	11419	40.69367	-73.81783	"(40.69367, -73.81783)"
	8/23/2016	10:50			40.710945	-74.01441	"(40.710945, -74.01441)"

Count - select COUNT(*) from finaldamgproject.stg_nyc;



The screenshot shows the MySQL Workbench interface with a query editor and a result grid. The query editor contains the following code:

```
7 #Austin
8 • select * from finaldamgproject.stg_austin;
9 • select count(*) from finaldamgproject.stg_austin;
10 • select * from finaldamgproject.transform_austin;
11 • select count(*) from finaldamgproject.transform_austin;
12
13 #NewYork
14 • select * from finaldamgproject.stg_nyc;
15 • select COUNT(*) from finaldamgproject.stg_nyc;
16 • select * from finaldamgproject.transform_nyc;
17 • select count(*) from finaldamgproject.transform_nyc;
18
19 #Chicago
20 • select * from finaldamgproject.stg_chicago;
21 • select COUNT(*) from finaldamgproject.stg_chicago;
```

The result grid displays the following data:

	COUNT(*)
▶	1048575

DDL commands

```
CREATE TABLE stg_nyc (
    CRASH_DATE varchar(100) DEFAULT NULL,
    CRASH_TIME varchar(100) DEFAULT NULL,
    BOROUGH varchar(100) DEFAULT NULL,
    ZIP_CODE varchar(100) DEFAULT NULL,
    LATITUDE varchar(100) DEFAULT NULL,
    LONGITUDE varchar(100) DEFAULT NULL,
    LOCATION varchar(100) DEFAULT NULL,
    ON_STREET_NAME varchar(100) DEFAULT NULL,
    CROSS_STREET_NAME varchar(100) DEFAULT NULL,
    OFF_STREET_NAME varchar(100) DEFAULT NULL,
    NUMBER_OF_PERSONS_INJURED varchar(100) DEFAULT NULL,
    NUMBER_OF_PERSONS_KILLED int DEFAULT NULL,
    NUMBER_OF_PEDESTRIANS_INJURED int DEFAULT NULL,
    NUMBER_OF_PEDESTRIANS_KILLED int DEFAULT NULL,
    NUMBER_OF_CYCLIST_INJURED varchar(100) DEFAULT NULL,
    NUMBER_OF_CYCLIST_KILLED varchar(100) DEFAULT NULL,
    NUMBER_OF_MOTORIST_INJURED varchar(100) DEFAULT NULL,
    NUMBER_OF_MOTORIST_KILLED varchar(100) DEFAULT NULL,
    CONTRIBUTING_FACTOR_VEHICLE_1 varchar(100) DEFAULT
NULL,
    CONTRIBUTING_FACTOR_VEHICLE_2 varchar(100) DEFAULT
NULL,
    CONTRIBUTING_FACTOR_VEHICLE_3 varchar(100) DEFAULT
NULL,
    CONTRIBUTING_FACTOR_VEHICLE_4 varchar(100) DEFAULT
NULL,
    CONTRIBUTING_FACTOR_VEHICLE_5 varchar(100) DEFAULT
NULL,
    COLLISION_ID varchar(100) DEFAULT NULL,
    VEHICLE_TYPE_CODE_1 varchar(100) DEFAULT NULL,
    VEHICLE_TYPE_CODE_2 varchar(100) DEFAULT NULL,
    VEHICLE_TYPE_CODE_3 varchar(100) DEFAULT NULL,
    VEHICLE_TYPE_CODE_4 varchar(100) DEFAULT NULL,
```

```

VEHICLE_TYPE_CODE_5 varchar(100) DEFAULT NULL,
DI_CreatedDate datetime DEFAULT NULL,
DI_WorkFlowFileName varchar(100) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci

```

ContributingFactor

Stage Table: Stg_ContributingFactor_Dataset

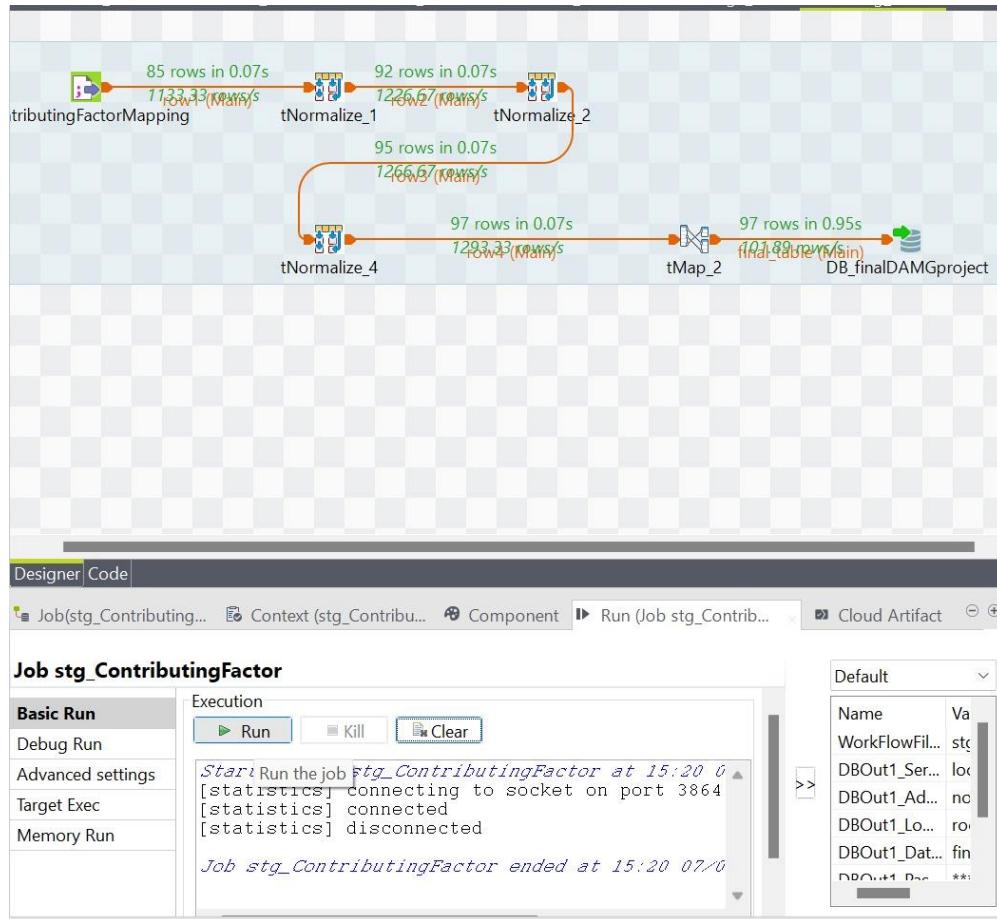
Purpose: Staging Contributing Factor Dataset from tsv file

tNormalize1: Chicago - (;)

tNormalize2: NYC - (;)

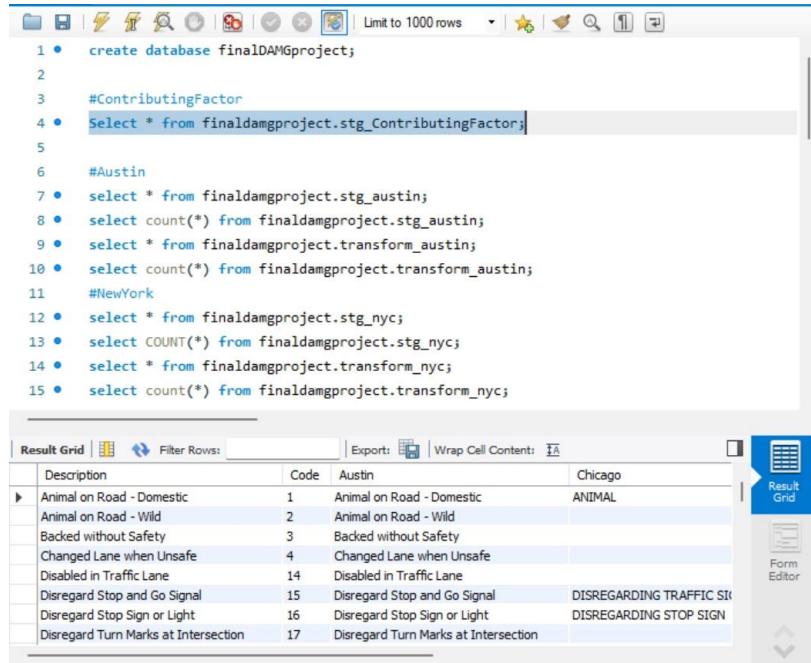
tNormalize4: NYC - (,)

tMAP >> trimming



MYSQL Command

Table - Select * from finaldamgproject.stg_ContributingFactor;

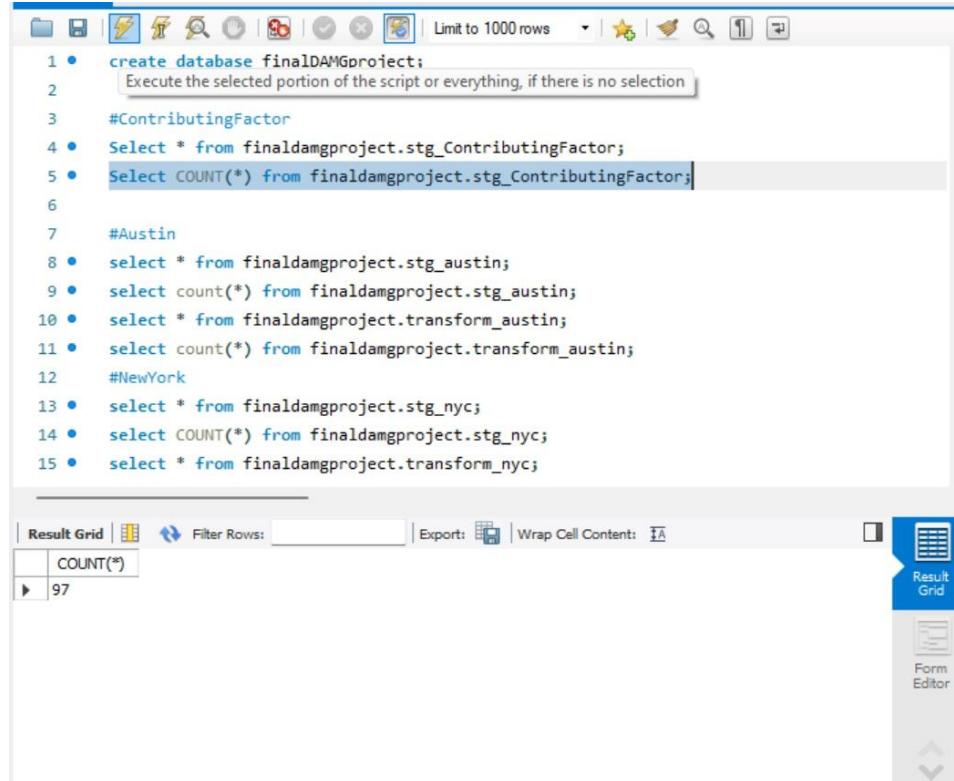


```
1 • create database finalDAMGproject;
2
3 • #ContributingFactor
4 • Select * from finaldamgproject.stg_ContributingFactor;
5
6 • #Austin
7 • select * from finaldamgproject.stg_austin;
8 • select count(*) from finaldamgproject.stg_austin;
9 • select * from finaldamgproject.transform_austin;
10 • select count(*) from finaldamgproject.transform_austin;
11 • #NewYork
12 • select * from finaldamgproject.stg_nyc;
13 • select COUNT(*) from finaldamgproject.stg_nyc;
14 • select * from finaldamgproject.transform_nyc;
15 • select count(*) from finaldamgproject.transform_nyc;
```

The screenshot shows the MySQL Workbench interface with a query editor containing the provided SQL code. Below the editor is a result grid pane displaying the data from the stg_ContributingFactor table. The table has four columns: Description, Code, Austin, and Chicago. The data includes various contributing factors like 'Animal on Road - Domestic' and 'Backed without Safety', each assigned a unique code (1-17) and mapped to specific categories in the Austin and Chicago columns.

Description	Code	Austin	Chicago
Animal on Road - Domestic	1	Animal on Road - Domestic	ANIMAL
Animal on Road - Wild	2	Animal on Road - Wild	
Backed without Safety	3	Backed without Safety	
Changed Lane when Unsafe	4	Changed Lane when Unsafe	
Disabled in Traffic Lane	14	Disabled in Traffic Lane	
Disregard Stop and Go Signal	15	Disregard Stop and Go Signal	DISREGARDING TRAFFIC SIG
Disregard Stop Sign or Light	16	Disregard Stop Sign or Light	DISREGARDING STOP SIGN
Disregard Turn Marks at Intersection	17	Disregard Turn Marks at Intersection	

Count - Select COUNT(*) from finaldamgproject.stg_ContributingFactor;



```
1 • create database finalDAMGproject;
   Execute the selected portion of the script or everything, if there is no selection
2
3 • #ContributingFactor
4 • Select * from finaldamgproject.stg_ContributingFactor;
5 • Select COUNT(*) from finaldamgproject.stg_ContributingFactor;
6
7 • #Austin
8 • select * from finaldamgproject.stg_austin;
9 • select count(*) from finaldamgproject.stg_austin;
10 • select * from finaldamgproject.transform_austin;
11 • select count(*) from finaldamgproject.transform_austin;
12 • #NewYork
13 • select * from finaldamgproject.stg_nyc;
14 • select COUNT(*) from finaldamgproject.stg_nyc;
15 • select * from finaldamgproject.transform_nyc;
```

The screenshot shows the MySQL Workbench interface with a query editor containing the provided SQL code. Below the editor is a result grid pane displaying the count of rows from the stg_ContributingFactor table. The result is a single row with the value '97' in the COUNT(*) column.

COUNT(*)
97

DDL commands

```
CREATE TABLE `stg_contributingfactor` (
  `Description` varchar(1000) DEFAULT NULL,
  `Code` int DEFAULT NULL,
  `Austin` varchar(200) DEFAULT NULL,
  `Chicago` varchar(200) DEFAULT NULL,
  `New_York` varchar(200) DEFAULT NULL,
  `DI_CreatedDate` datetime DEFAULT NULL,
  `DI_WorkFlowFileName` varchar(100) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci
```

Transform

Austin

Transform Table: Transforming_Austin

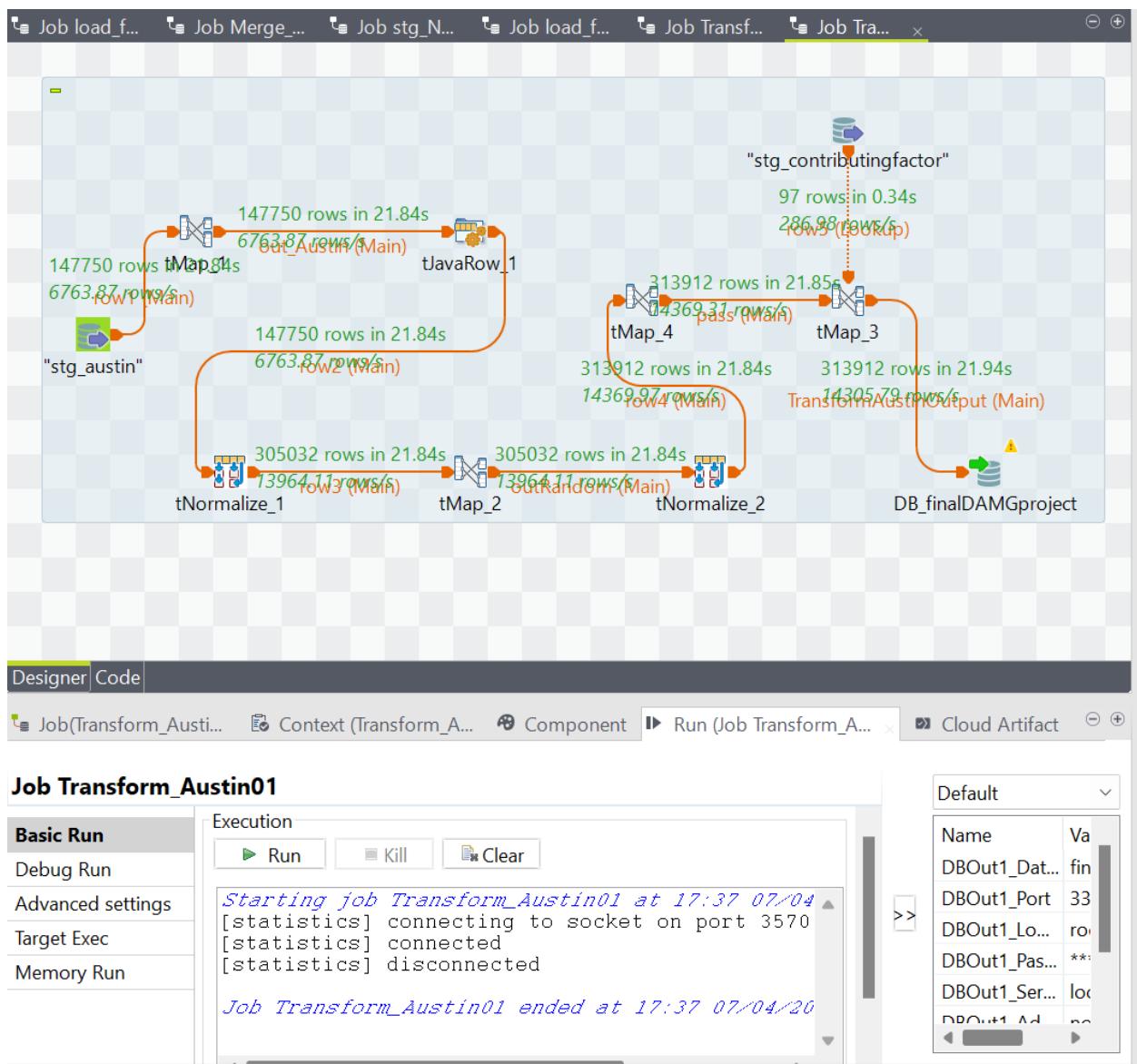
Purpose

tJavaRow: Calculate total vehicle count

tNormalize: normalize Vehicle Type

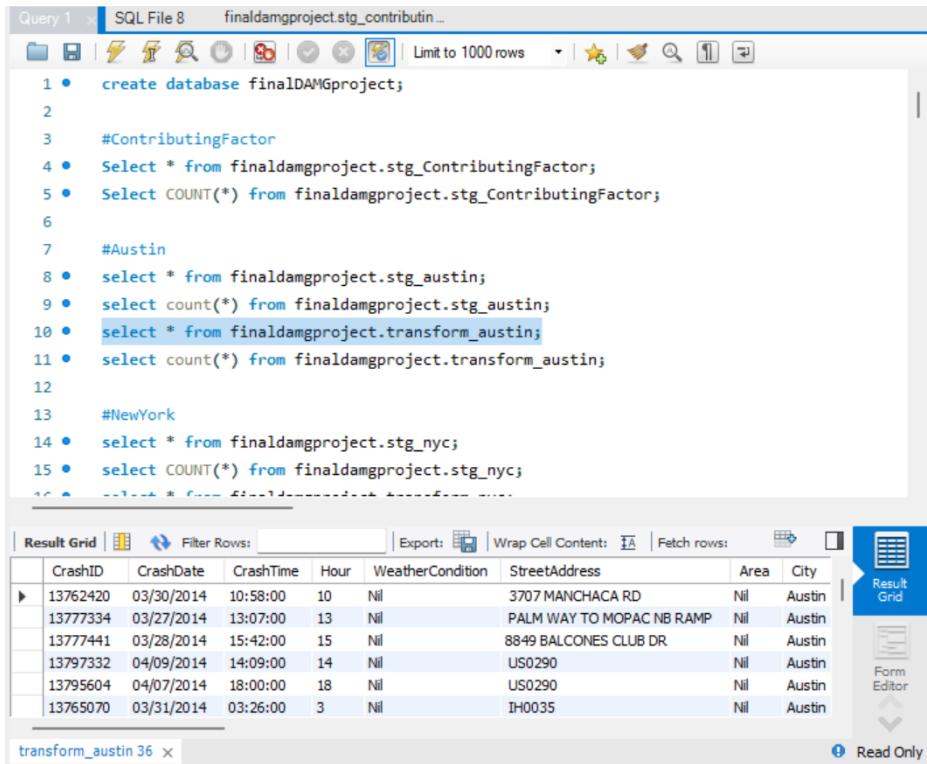
tNormalize: Factor Code

tMAP: joining Austin to contributing factor to get Description



MYSQL Command

Table



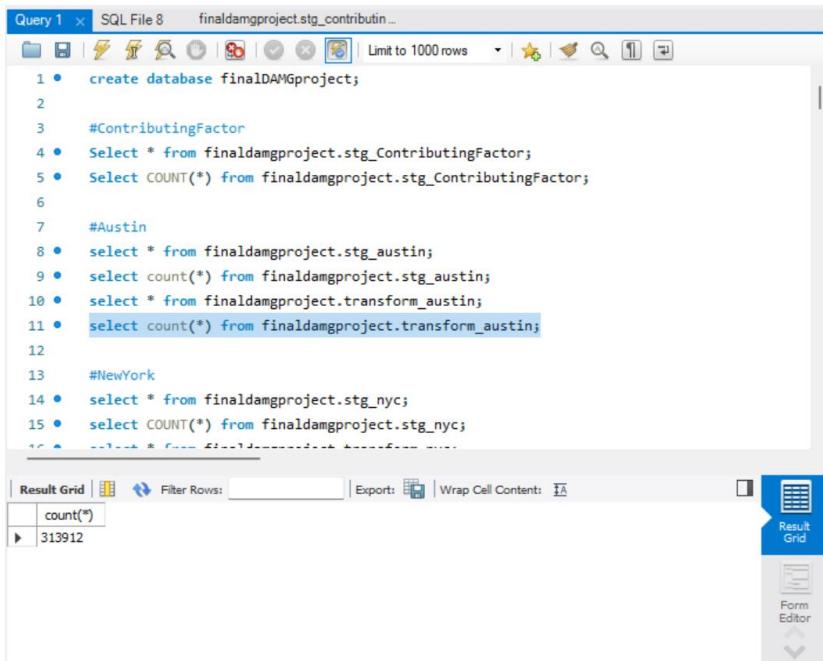
The screenshot shows the MySQL Workbench interface with a query editor and a results grid. The query editor contains the following SQL code:

```
Query 1 | SQL File 8 finaldamgproject.stg_contributin...
1 • create database finalDAMGproject;
2
3 #ContributingFactor
4 • Select * from finaldamgproject.stg_ContributingFactor;
5 • Select COUNT(*) from finaldamgproject.stg_ContributingFactor;
6
7 #Austin
8 • select * from finaldamgproject.stg_austin;
9 • select count(*) from finaldamgproject.stg_austin;
10 • select * from finaldamgproject.transform_austin;
11 • select count(*) from finaldamgproject.transform_austin;
12
13 #NewYork
14 • select * from finaldamgproject.stg_nyc;
15 • select COUNT(*) from finaldamgproject.stg_nyc;
```

The results grid displays data from the 'transform_austin' table:

CrashID	CrashDate	CrashTime	Hour	WeatherCondition	StreetAddress	Area	City
13762420	03/30/2014	10:58:00	10	Nil	3707 MANCHACA RD	Nil	Austin
13777334	03/27/2014	13:07:00	13	Nil	PALM WAY TO MOPAC NB RAMP	Nil	Austin
13777441	03/28/2014	15:42:00	15	Nil	8849 BALCONES CLUB DR	Nil	Austin
13797332	04/09/2014	14:09:00	14	Nil	US0290	Nil	Austin
13795604	04/07/2014	18:00:00	18	Nil	US0290	Nil	Austin
13765070	03/31/2014	03:26:00	3	Nil	IH0035	Nil	Austin

Count



The screenshot shows the MySQL Workbench interface with a query editor and a results grid. The query editor contains the following SQL code:

```
Query 1 | SQL File 8 finaldamgproject.stg_contributin...
1 • create database finalDAMGproject;
2
3 #ContributingFactor
4 • Select * from finaldamgproject.stg_ContributingFactor;
5 • Select COUNT(*) from finaldamgproject.stg_ContributingFactor;
6
7 #Austin
8 • select * from finaldamgproject.stg_austin;
9 • select count(*) from finaldamgproject.stg_austin;
10 • select * from finaldamgproject.transform_austin;
11 • select count(*) from finaldamgproject.transform_austin;
12
13 #NewYork
14 • select * from finaldamgproject.stg_nyc;
15 • select COUNT(*) from finaldamgproject.stg_nyc;
```

The results grid displays the result of the COUNT(*) query:

count(*)
313912

DDL commands

```
CREATE TABLE `transform_austin` (
  `CrashID` varchar(500) DEFAULT NULL,
  `CrashDate` varchar(20) DEFAULT NULL,
  `CrashTime` varchar(10) DEFAULT NULL,
  `Hour` int DEFAULT NULL,
  `WeatherCondition` varchar(100) DEFAULT NULL,
  `StreetAddress` varchar(500) DEFAULT NULL,
  `Area` varchar(20) DEFAULT NULL,
  `City` varchar(20) DEFAULT NULL,
  `ZipCode` varchar(10) DEFAULT NULL,
  `Latitude` varchar(20) DEFAULT NULL,
  `Longitude` varchar(20) DEFAULT NULL,
  `SourceName` varchar(50) DEFAULT NULL,
  `MotoristKilled` int DEFAULT NULL,
  `MotoristInjured` int DEFAULT NULL,
  `PedestriansKilled` int DEFAULT NULL,
  `PedestriansInjured` int DEFAULT NULL,
  `CyclistKilled` int DEFAULT NULL,
  `CyclistInjured` int DEFAULT NULL,
  `TotalKilled` int DEFAULT NULL,
  `TotalInjured` int DEFAULT NULL,
  `VehicleType` varchar(250) DEFAULT NULL,
  `VehicleCount` int DEFAULT NULL,
  `FactorCode` int DEFAULT NULL,
  `FactorDescription` varchar(1000) DEFAULT NULL,
  `DI_CreatedDate` datetime DEFAULT NULL,
  `DI_WorkFlowFileName` varchar(100) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci
```

Chicago

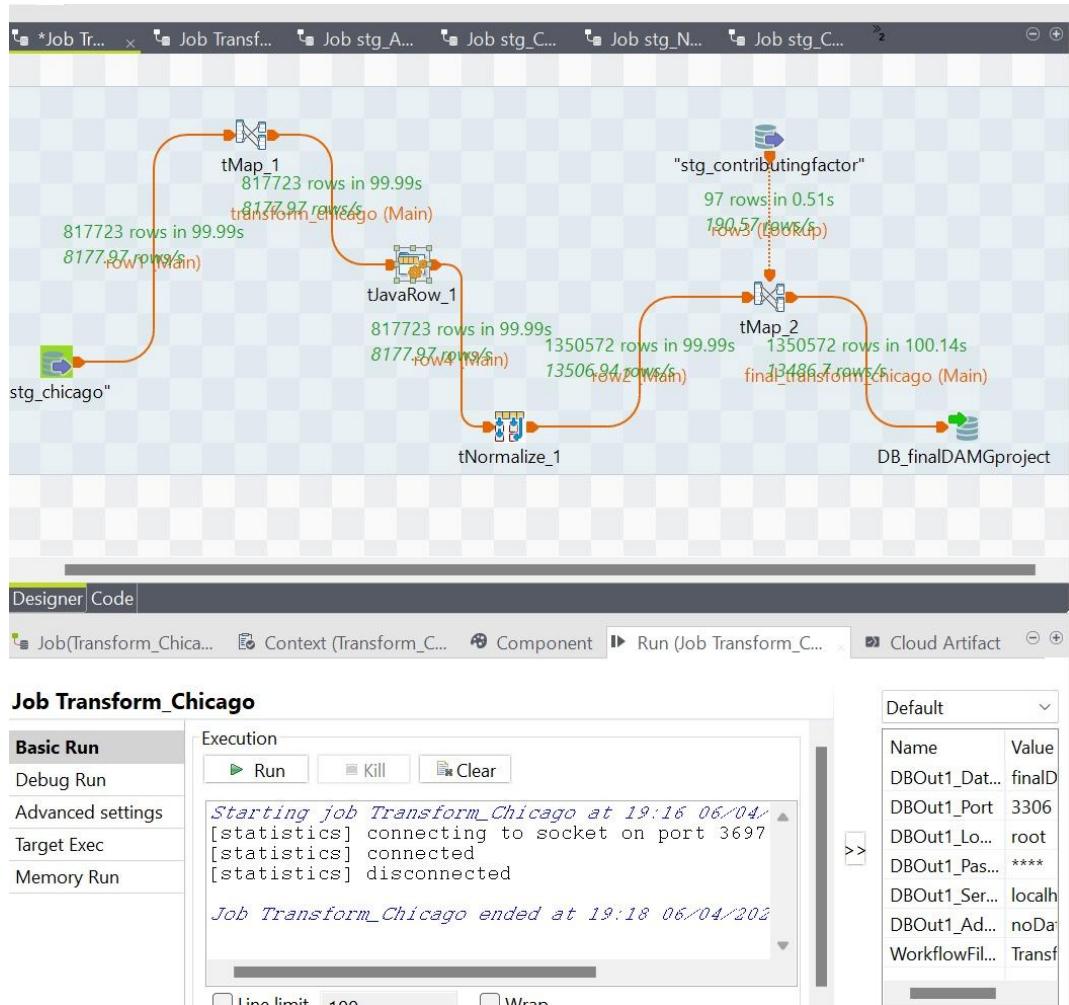
Transform Table: Transforming_Chicago

Purpose

tJavaRow: Denormalize factor description

tNormalize: Factor Description

tMAP: joining Chicago to contributing factor to get Code



MYSQL Command

Table - select * from finaldamgproject.transform_chicago;

The screenshot shows the MySQL Workbench interface with a script editor and a result grid. The script editor contains a series of SQL commands, mostly comments starting with '#', interspersed with actual queries. The last query in the script is:

```
17 • select * from finaldamgproject.transform_chicago;
```

The result grid below displays the data from the transform_chicago table. The columns are: CrashID, CrashDate, CrashTime, Hour, WeatherCondition, and StreetAddress. The data consists of approximately 10 rows of crash information.

CrashID	CrashDate	CrashTime	Hour	WeatherCondition	StreetAddress
6c1659069e9c6285a650e70d6f9b574ed5f64c1...	08/18/2023	12:50:00	12	CLEAR	700 OHARE S...
6c1659069e9c6285a650e70d6f9b574ed5f64c1...	08/18/2023	12:50:00	12	CLEAR	700 OHARE S...
5f54a59fc087b12ae5b1acff96a3caf4f2d37e7...	07/29/2023	14:45:00	14	CLEAR	2101 ASHLAN...
5f54a59fc087b12ae5b1acff96a3caf4f2d37e7...	07/29/2023	14:45:00	14	CLEAR	2101 ASHLAN...
61fcb8c1eb522a6469b460e2134df3d15f82e81f...	08/18/2023	17:58:00	17	CLEAR	3422 LONG A...
61fcb8c1eb522a6469b460e2134df3d15f82e81f...	08/18/2023	17:58:00	17	CLEAR	3422 LONG A...
004cd14d0303a9163aad69a2d7f341b7da2a85...	11/26/2019	08:38:00	8	CLEAR	5 TERMINAL ...
nnn-n14-n3n?n=016?n=7A7F241h7d=7=85	11/26/2019	08:38:00	8	CLEAR	5 TERMINAL ...

Count - select COUNT(*) from finaldamgproject.transform_chicago;

The screenshot shows the MySQL Workbench interface with a script editor and a result grid. The script editor contains a series of SQL commands, mostly comments starting with '#', interspersed with actual queries. The last query in the script is:

```
18 • select COUNT(*) from finaldamgproject.transform_chicago;
```

The result grid below displays the count of rows. The data consists of a single row with the value 1350572.

COUNT(*)
1350572

DDL commands

```
CREATE TABLE transform_chicago (
    CrashID varchar(500) DEFAULT NULL,
    CrashDate varchar(20) DEFAULT NULL,
    CrashTime varchar(10) DEFAULT NULL,
    Hour int DEFAULT NULL,
    WeatherCondition varchar(100) DEFAULT NULL,
    StreetAddress varchar(500) DEFAULT NULL,
    Area varchar(20) DEFAULT NULL,
    City varchar(20) DEFAULT NULL,
    ZipCode varchar(10) DEFAULT NULL,
    Latitude varchar(20) DEFAULT NULL,
    Longitude varchar(20) DEFAULT NULL,
    SourceName varchar(50) DEFAULT NULL,
    MotoristKilled int DEFAULT NULL,
    MotoristInjured int DEFAULT NULL,
    PedastriansKilled int DEFAULT NULL,
    PedastriansInjured int DEFAULT NULL,
    CyclistKilled int DEFAULT NULL,
    CyclistInjured int DEFAULT NULL,
    TotalKilled int DEFAULT NULL,
    TotalInjured int DEFAULT NULL,
    VehicleType varchar(250) DEFAULT NULL,
    VehicleCount int DEFAULT NULL,
    FactorCode int DEFAULT NULL,
    FactorDescription varchar(1000) DEFAULT NULL,
    DI_CreatedDate datetime DEFAULT NULL,
    DI_WorkFlowFileName varchar(100) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci
```

NewYork

Transform Table: Transforming_NYC

Purpose

tFilterRow: 2 Rows Rejecting

Collision_ID : empty – 2 rows

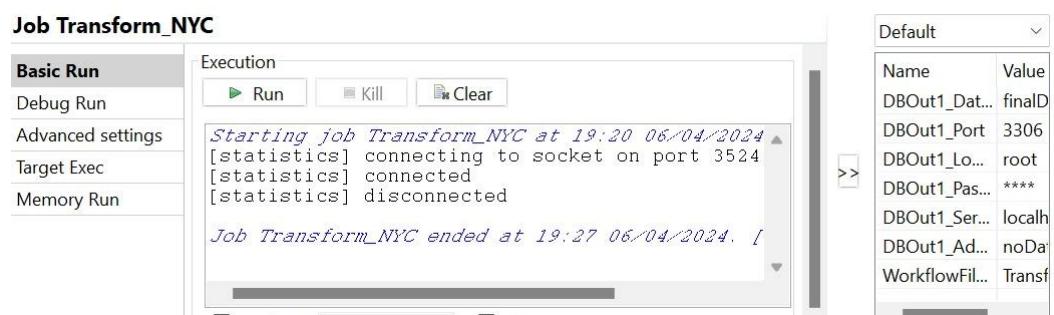
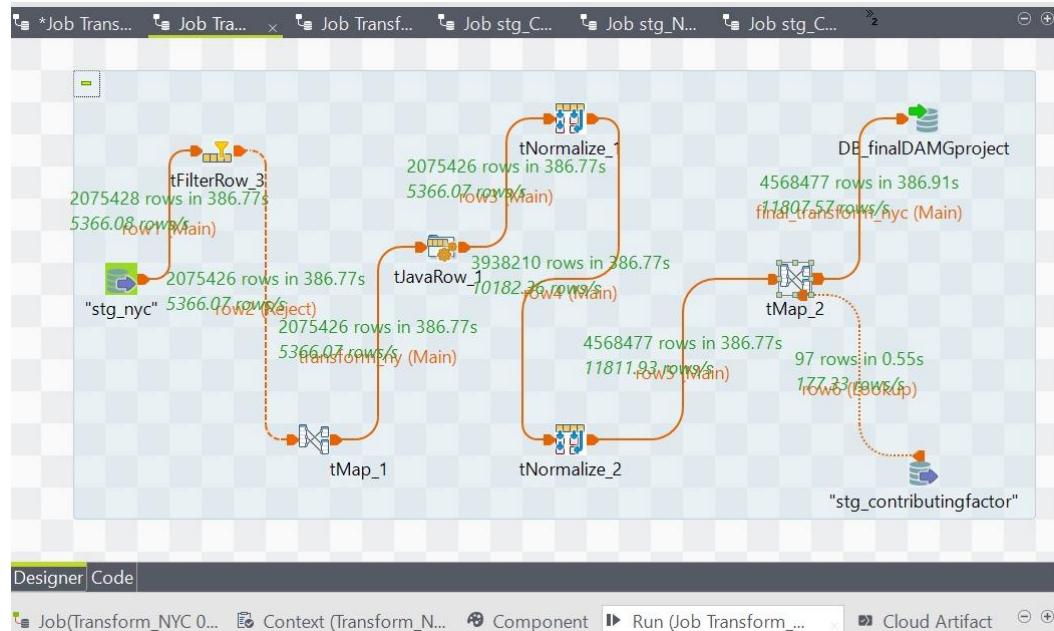
Number of Cyclist – Passenger Vehicle – Dirty info

tJavaRow: Denormalize Vehicle Type and Factor Description

tNormalize: normalize Vehicle Type

tNormalize: Factor Factor Description

tMAP: joining NYC to contributing factor



MYSQL Command

Table - select * from finaldamgproject.transform_nyc;

The screenshot shows the MySQL Workbench interface with a query editor titled "Query 1" containing the following SQL code:

```
4 •  select * from finaldamgproject.stg_austin;
5 •  select count(*) from finaldamgproject.stg_austin;
6 •  select * from finaldamgproject.transform_austin;
7 •  select count(*) from finaldamgproject.transform_austin;
8      #NewYork
9 •  select * from finaldamgproject.stg_nyc;
10 •  select COUNT(*) from finaldamgproject.stg_nyc;
11 •  select * from finaldamgproject.transform_nyc;
12 •  select count(*) from finaldamgproject.transform_nyc;
13
14      #Chicago
```

Below the code, the "Result Grid" tab is selected, displaying the following data:

CrashID	CrashDate	CrashTime	Hour	WeatherCondition	StreetAddress
4455765	09/11/2021	02:39:00	2	Nil	ON - WHITESTONE EXPRESSWAYCROSS - 20 A...
4455765	09/11/2021	02:39:00	2	Nil	ON - WHITESTONE EXPRESSWAYCROSS - 20 A...
4513547	03/26/2022	11:45:00	11	Nil	ON - QUEENSBORO BRIDGE UPPERCROSS - OFF...
4541903	06/29/2022	06:55:00	6	Nil	ON - THROGS NECK BRIDGE CROSS - OFF -
4541903	06/29/2022	06:55:00	6	Nil	ON - THROGS NECK BRIDGE CROSS - OFF -
4456314	09/11/2021	09:35:00	9	Nil	ON - CROSS - OFF - 1211 LORING AVENUE E
4486609	12/14/2021	08:13:00	8	Nil	ON - SARATOGA AVENUECROSS - DECATUR ST... E
4407458	04/14/2021	12:47:00	12	Nil	ON - MAJOR DEEGAN EXPRESSWAY RAMPCRO...
4407458	04/14/2021	12:47:00	12	Nil	ON - MAJOR DEEGAN EXPRESSWAY RAMPCRO...
4486555	12/14/2021	17:05:00	17	Nil	ON - BROOKLYN QUEENS EXPRESSWAYCROSS ...
4486555	12/14/2021	17:05:00	17	Nil	ON - BROOKLYN QUEENS EXPRESSWAYCROSS ...
4486660	12/14/2021	08:17:00	8	Nil	ON - CROSS - OFF - 344 BAYCHESTER AVE... E
4486660	12/14/2021	08:17:00	8	Nil	ON - CROSS - OFF - 344 BAYCHESTER AVE... E

Count - select count(*) from finaldamgproject.transform_nyc;

The screenshot shows the MySQL Workbench interface with a query editor titled "Query 1" containing the following SQL code:

```
1 •  create database finalDAMGproject;
2
3      #Austin
4 •  select * from finaldamgproject.stg_austin;
5 •  select count(*) from finaldamgproject.stg_austin;
6 •  select * from finaldamgproject.transform_austin;
7 •  select count(*) from finaldamgproject.transform_austin;
8      #NewYork
9 •  select * from finaldamgproject.stg_nyc;
10 •  select COUNT(*) from finaldamgproject.stg_nyc;
11 •  select * from finaldamgproject.transform_nyc;
12 •  select count(*) from finaldamgproject.transform_nyc;
13
14      #Chicago
15 •  select * from finaldamgproject.stg_chicago;
```

Below the code, the "Result Grid" tab is selected, displaying the following data:

count(*)
4568477

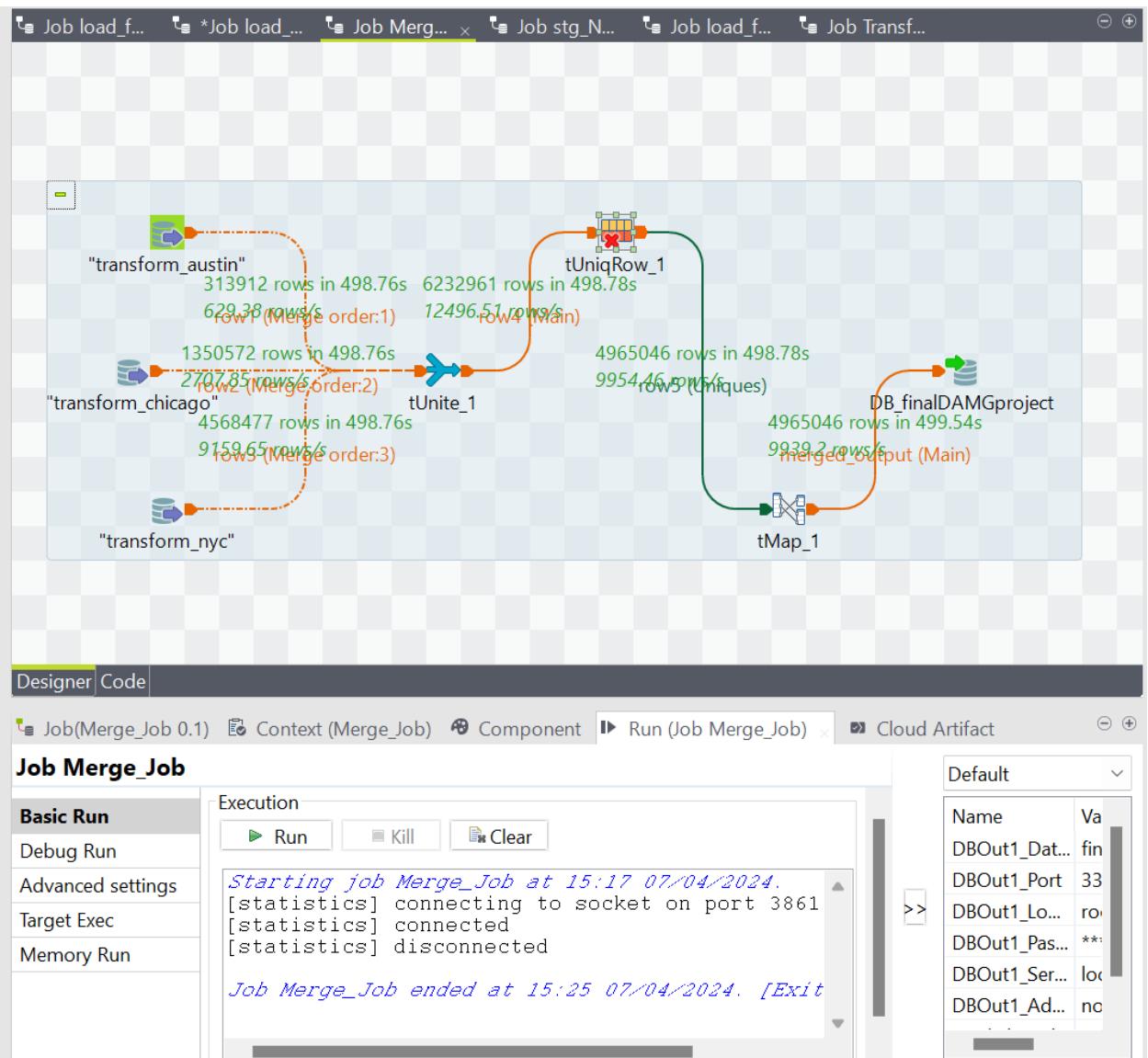
DDL commands

```
CREATE TABLE transform_nyc (
    CrashID varchar(500) DEFAULT NULL,
    CrashDate varchar(20) DEFAULT NULL,
    CrashTime varchar(10) DEFAULT NULL,
    Hour int DEFAULT NULL,
    WeatherCondition varchar(100) DEFAULT NULL,
    StreetAddress varchar(500) DEFAULT NULL,
    Area varchar(20) DEFAULT NULL,
    City varchar(20) DEFAULT NULL,
    ZipCode varchar(10) DEFAULT NULL,
    Latitude varchar(20) DEFAULT NULL,
    Longitude varchar(20) DEFAULT NULL,
    SourceName varchar(50) DEFAULT NULL,
    MotoristKilled int DEFAULT NULL,
    MotoristInjured int DEFAULT NULL,
    PedastriansKilled int DEFAULT NULL,
    PedastriansInjured int DEFAULT NULL,
    CyclistKilled int DEFAULT NULL,
    CyclistInjured int DEFAULT NULL,
    TotalKilled int DEFAULT NULL,
    TotalInjured int DEFAULT NULL,
    VehicleType varchar(250) DEFAULT NULL,
    VehicleCount int DEFAULT NULL,
    FactorCode int DEFAULT NULL,
    FactorDescription varchar(1000) DEFAULT NULL,
    DI_CreatedDate datetime DEFAULT NULL,
    DI_WorkFlowFileName varchar(100) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci
```

Merge

Merging Table: Merge_Job

Purpose: Uniting transform_austin, transform_chicago, transform_nyc



MYSQL Command

Table : select * from finaldamgproject.merged_report;

The screenshot shows the MySQL Workbench interface with a query editor and a result grid. The query editor contains the following SQL code:

```
Query 1 | SQL File 8 | finaldamgproject.stg_contributin...
24
25  #Merged
26 • select * from finaldamgproject.merged_report;
27 • select COUNT(*) from finaldamgproject.merged_report;
28 • select * from finaldamgproject.merged_report where City='Austin';
29 • select * from finaldamgproject.merged_report where City ='Chicago';
30 • select * from finaldamgproject.merged_report where City='New York';
31
32
33 • create database finalProject_DataWarehouse;
34
35  #Date Dimension
36 • select * from finalProject_DataWarehouse.Dim_Date;
37
38  #Time Dimension
```

The result grid displays the following data:

CrashID	CrashDate	CrashTime	Hour	WeatherCondition	StreetAddress	Area
13762420	2014-03-30 00:00:00	10:58:00	10	Nil	3707 MANCHACA RD	Nil
13777334	2014-03-27 00:00:00	13:07:00	13	Nil	PALM WAY TO MOPAC NB RAMP	Nil
13777441	2014-03-28 00:00:00	15:42:00	15	Nil	8849 BALCONES CLUB DR	Nil
13797332	2014-04-09 00:00:00	14:09:00	14	Nil	US0290	Nil
13795604	2014-04-07 00:00:00	18:00:00	18	Nil	US0290	Nil
13765070	2014-03-31 00:00:00	03:26:00	3	Nil	IH0035	Nil

Count : select COUNT(*) from finaldamgproject.merged_report;

The screenshot shows the MySQL Workbench interface with a query editor and a result grid. The query editor contains the following SQL code:

```
Query 1 | SQL File 8 | finaldamgproject.stg_contributin...
24
25  #Merged
26 • select * from finaldamgproject.merged_report;
27 • select COUNT(*) from finaldamgproject.merged_report;
28 • select * from finaldamgproject.merged_report where City='Austin';
29 • select * from finaldamgproject.merged_report where City ='Chicago';
30 • select * from finaldamgproject.merged_report where City='New York';
31
32
33 • create database finalProject_DataWarehouse;
34
35  #Date Dimension
36 • select * from finalProject_DataWarehouse.Dim_Date;
37
38  #Time Dimension
```

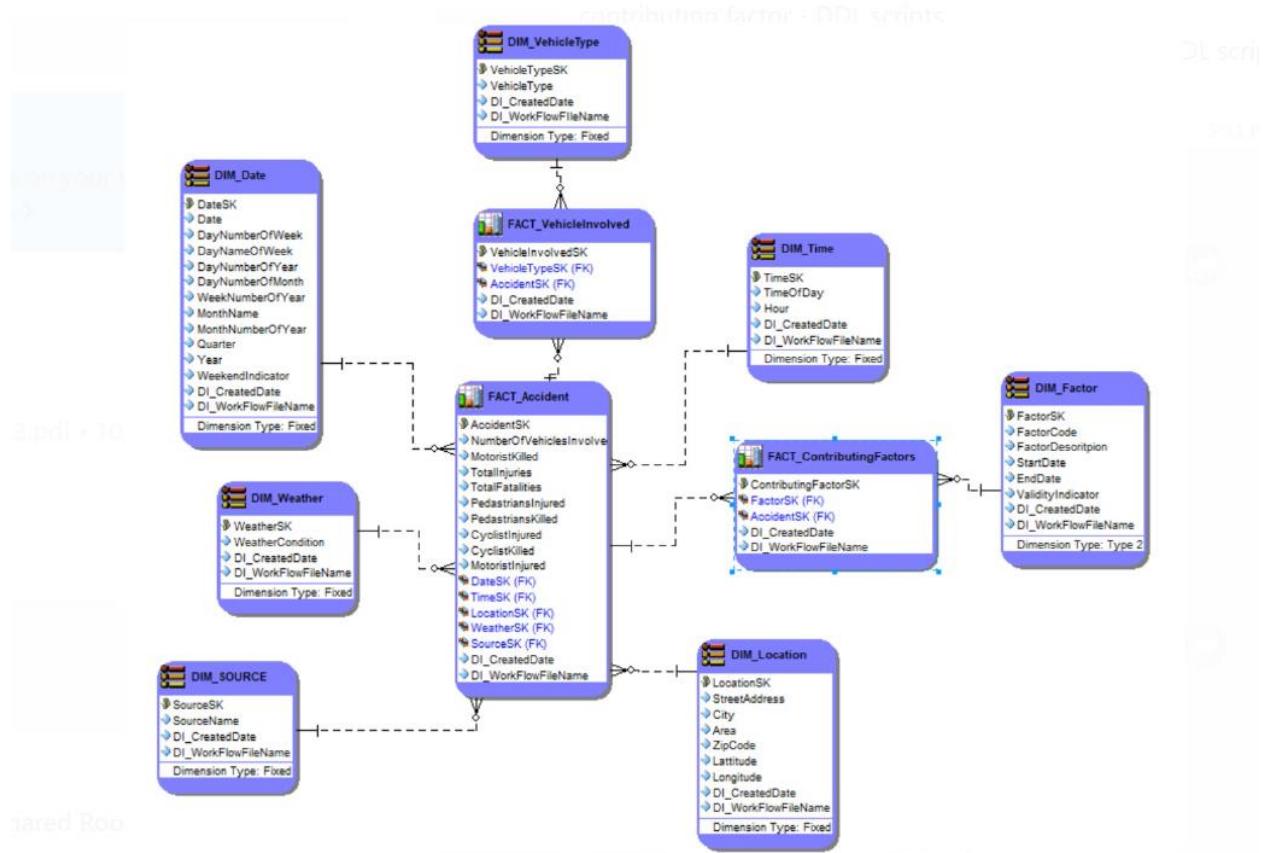
The result grid displays the following data:

COUNT(*)
4965046

DDL commands

```
CREATE TABLE merged_report (
    CrashID varchar(500) DEFAULT NULL,
    CrashDate varchar(20) DEFAULT NULL,
    CrashTime varchar(10) DEFAULT NULL,
    Hour int DEFAULT NULL,
    WeatherCondition varchar(100) DEFAULT NULL,
    StreetAddress varchar(500) DEFAULT NULL,
    Area varchar(20) DEFAULT NULL,
    City varchar(20) DEFAULT NULL,
    ZipCode varchar(10) DEFAULT NULL,
    Latitude varchar(20) DEFAULT NULL,
    Longitude varchar(20) DEFAULT NULL,
    SourceName varchar(50) DEFAULT NULL,
    MotoristKilled int DEFAULT NULL,
    MotoristInjured int DEFAULT NULL,
    PedastriansKilled int DEFAULT NULL,
    PedastriansInjured int DEFAULT NULL,
    CyclistKilled int DEFAULT NULL,
    CyclistInjured int DEFAULT NULL,
    TotalKilled int DEFAULT NULL,
    TotalInjured int DEFAULT NULL,
    VehicleType varchar(250) DEFAULT NULL,
    VehicleCount int DEFAULT NULL,
    FactorCode int DEFAULT NULL,
    FactorDescription varchar(1000) DEFAULT NULL,
    DI_CreatedDate datetime DEFAULT NULL,
    DI_WorkFlowFileName varchar(100) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci
```

Dimensional Model



Dimension Tables:

- **DIM_Date**: Contains attributes like DateSK, Date, DayNumberOfWeek, DayNameOfWeek, DayNumberOfYear, MonthName, Quarter, Year, WeekendIndicator, DI_CreatedDate, DI_WorkFlowFileName.
- **DIM_Time**: Includes TimeSK, TimeOfDay, Hour, DI_CreatedDate, DI_WorkFlowFileName.
- **DIM_Weather**: Holds WeatherSK, WeatherCondition, DI_CreatedDate, DI_WorkFlowFileName.
- **DIM_Source**: Contains SourceSK, SourceName, DI_CreatedDate, DI_WorkFlowFileName.

- DIM_Location: Contains attributes like LocationSK, StreetAddress, Area, City, ZipCode, Latitude, Longitude, DI_CreatedDate, DI_WorkFlowFileName.
- DIM_VehicleType: Comprises VehicleTypeSK, VehicleType, DI_CreatedDate, DI_WorkFlowFileName.
- DIM_Factor - SCD Type 2 dimension - Contains FactorSK, FactorCode, FactorDescription, StartDate, EndDate, ValidityIndicator, DI_CreatedDate, DI_WorkFlowFileName.

Note: Choosing DIM_Factor as an SCD Type 2 is strategic because the nature of contributing factors to accidents may evolve over time, and the impact of these changes on accident analysis is significant. For example:

A contributing factor might be redefined or split into two separate factors, and analysts would need to distinguish accidents occurring under the old definition from those under the new one.

If a factor is no longer relevant, it can be ended without losing the historical data associated with it.

New factors might emerge, which would need to be tracked separately from the existing ones.

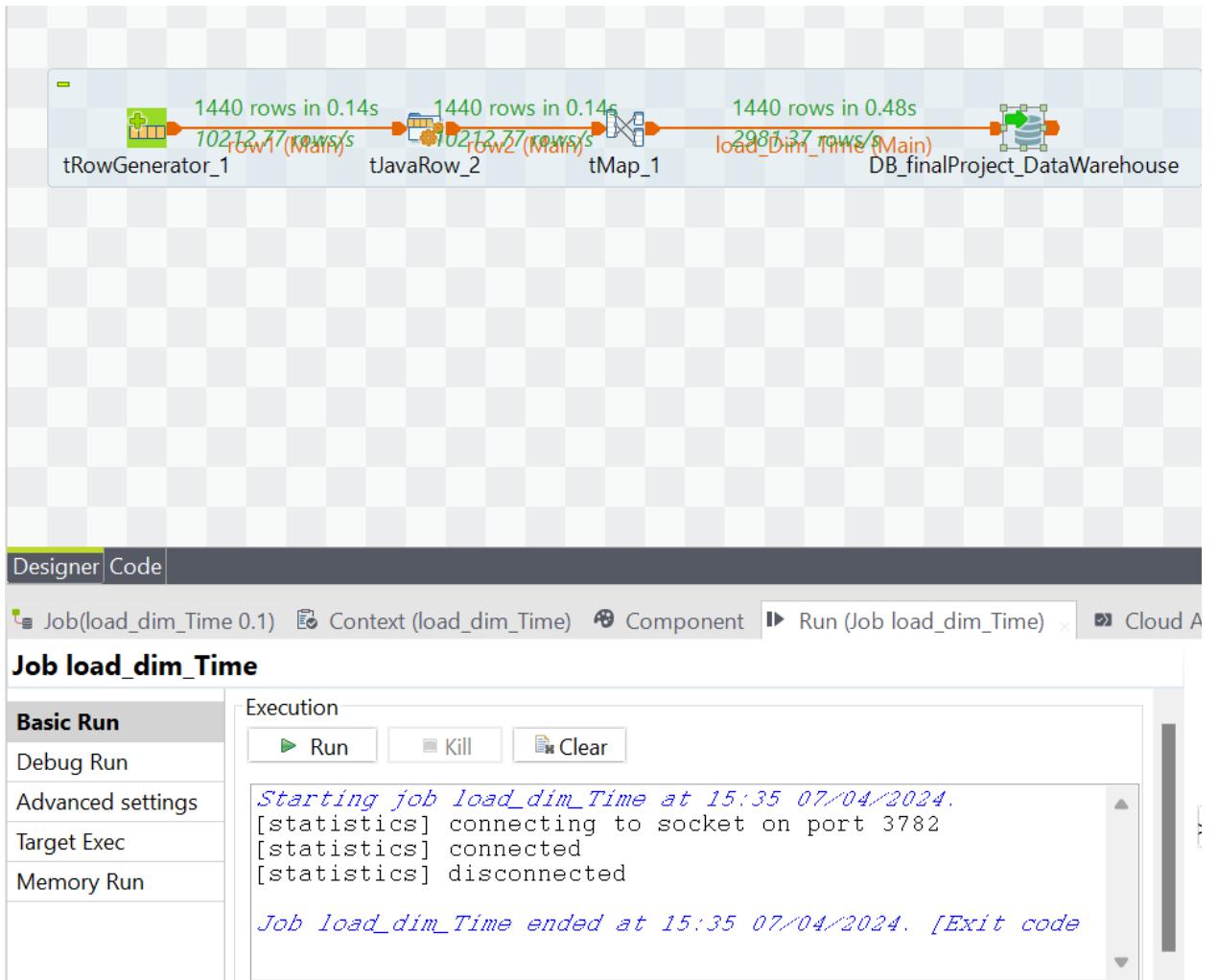
SCD Type 2 ensures that historical data integrity is maintained, and trends and analyses remain accurate even as the descriptions and relevancy of factors change over time. It allows the reporting to accurately reflect the context and details of contributing factors as they were at the time of each accident.

Fact Tables:

- FACT_Accident: Central fact table with AccidentSK, NumberofVehiclesInvolved, MotoristInjured, MotoristKilled, PedestriansInjured, PedestriansKilled, CyclistInjured, CyclistKilled, TotalInjured, TotalFatalities, DateSK, TimeSK, LocationSK, WeatherSK, SourceSK, DI_CreatedDate, DI_WorkFlowFileName.
- FACT_VehicleInvolved: Contains VehicleInvolvedSK, VehicleTypeSK, AccidentSK, DI_CreatedDate, DI_WorkFlowFileName.
- FACT_ContributingFactors: Includes ContributingFactorSK, FactorSK, AccidentSK, DI_CreatedDate, DI_WorkFlowFileName.

Finalproject_datawarehouse:

- Time Dimension



select * from finalProject_DataWarehouse.Dim_Time;

Navigator

Query 1 SQL File 8 finaldamgproject.stg_contributin... finaldamgproject.transform_aust... finalproject_datawa...

SCHEMAS

finaldamgproject
finalproject_datawarehouse

- Tables
- dim_date
- dim_factor
- dim_location
- dim_source
- dim_time
- dim_videotype
- dim_weather
- fact_accident
- fact_contributingfactors
- fact_vehicleinvolved
- Views
- Stored Procedures
- Functions
- jsonparsingloading
- pipe
- sakila
- sunflower
- sys
- tempd

Administration Schemas

Information

Table: dim_date

Columns:

DateSK	int PK	
Date	date	
DayNumberOfWeek	int	
20130101	2013-01-01	1
20130102	2013-01-02	2
20130103	2013-01-03	3
20130104	2013-01-04	4
20130105	2013-01-05	5
20130106	2013-01-06	6
20130107	2013-01-07	7
20130108	2013-01-08	8
20130109	2013-01-09	9
20130110	2013-01-10	10
20130111	2013-01-11	11
20130112	2013-01-12	12
20130113	2013-01-13	13
20130114	2013-01-14	14
20130115	2013-01-15	15
20130116	2013-01-16	16
20130117	2013-01-17	17
20130118	2013-01-18	18
20130119	2013-01-19	19

Dim_Date 56

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Result Grid | Form Editor | Field Types | Query Stats | Execution Plan | Apply | Revert

```
select COUNT(*) from finalProject_DataWarehouse.Dim_Time;
```

Query 1 SQL File 8 finaldamgproject.stg_contributin... finaldamgproject.transform_aust... finalproject_datawa...

38

39 #Time Dimension

40 • select * from finalProject_DataWarehouse.Dim_Time;

41 • select COUNT(*) from finalProject_DataWarehouse.Dim_Time;

42

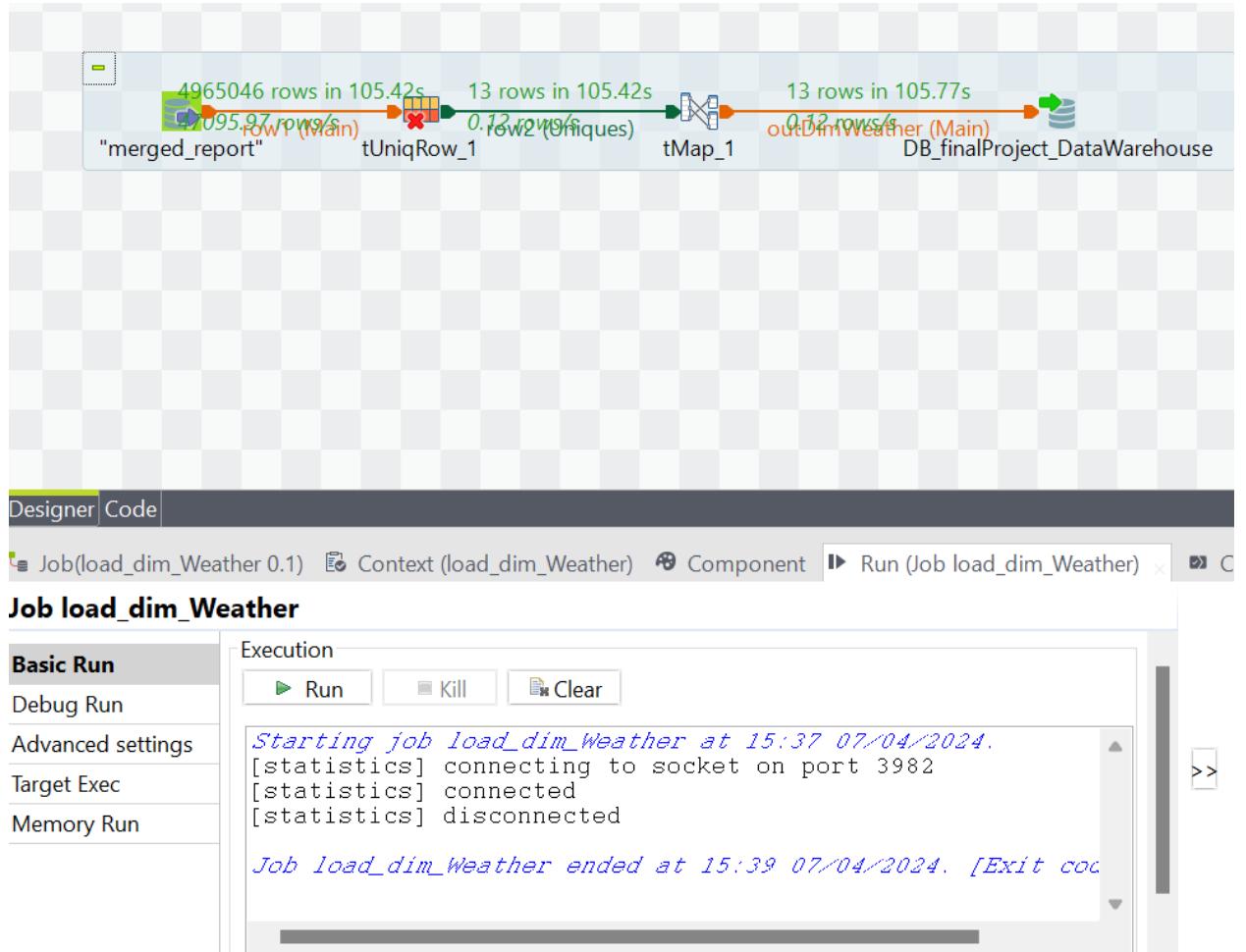
Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Result Grid | Form Editor

COUNT(*)
1440

```
CREATE TABLE `dim_time` (
`TimeSK` varchar(15) NOT NULL,
`TimeOfDay` varchar(15) DEFAULT NULL,
`Hour` int DEFAULT NULL,
`DI_CreatedDate` datetime DEFAULT NULL,
`DI_WorkflowFileName` varchar(50) DEFAULT NULL,
```

```
PRIMARY KEY(`TimeSK`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

- Weather Dimension



```
select * from finalProject_DataWarehouse.Dim_Weather;
```

Query 1 x SQL File 8 finaldamgproject.stg_contributin... finaldamgproject.transform_aust... finalproject_datawa...

42

43 #Weather Dimension

44 • select * from finalProject_DataWarehouse.Dim_Weather;

45 • select COUNT(*) from finalProject_DataWarehouse.Dim_Weather;

46

Result Grid | Filter Rows: Export: Wrap Cell Content: A

	WeatherSK	WeatherCondition	DI_CreatedDate	DI_WorkflowFileName
▶	1	Nil	2024-04-07 15:38:25	load_dim_weather
	2	CLEAR	2024-04-07 15:38:28	load_dim_weather
	3	SNOW	2024-04-07 15:38:28	load_dim_weather
	4	UNKNOWN	2024-04-07 15:38:28	load_dim_weather
	5	RAIN	2024-04-07 15:38:28	load_dim_weather
	6	CLOUDY/OVERCAST	2024-04-07 15:38:28	load_dim_weather
	7	FOG/SMOKE/HAZE	2024-04-07 15:38:28	load_dim_weather
	8	BLOWING SNOW	2024-04-07 15:38:28	load_dim_weather
	9	FREEZING RAIN/DRIZZLE	2024-04-07 15:38:28	load_dim_weather
	10	OTHER	2024-04-07 15:38:28	load_dim_weather
	11	SEVERE CROSS WIND G...	2024-04-07 15:38:28	load_dim_weather
	12	SLEET/HAIL	2024-04-07 15:38:28	load_dim_weather
	13	BLOWING SAND, SOIL, ...	2024-04-07 15:38:29	load_dim_weather

Result Grid Form Editor Field Types Query Stats Execution Plan

select COUNT(*) from finalProject_DataWarehouse.Dim_Weather;

The screenshot shows the MySQL Workbench interface. The top bar has tabs for 'Query 1' (selected), 'SQL File 8', and several database names. Below the tabs is a toolbar with various icons. The main area contains a SQL query:

```
42
43  #Weather Dimension
44 • select * from finalProject_DataWarehouse.Dim_Weather;
45 • select COUNT(*) from finalProject_DataWarehouse.Dim_Weather;
46
```

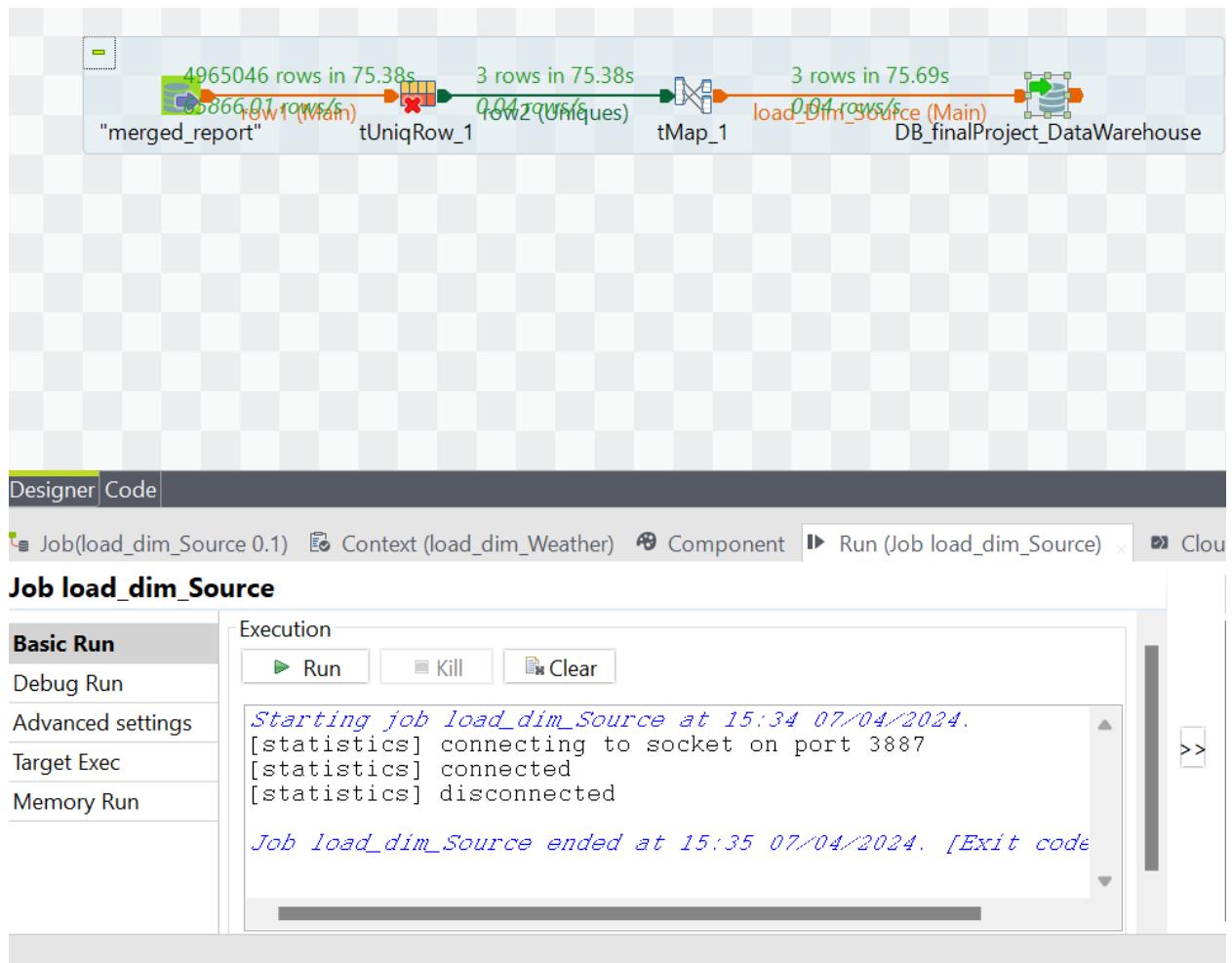
The second and third lines of the query are highlighted in blue. The result grid below shows the output of the last line:

COUNT(*)
13

On the right side of the results grid, there are two buttons: 'Result Grid' (highlighted in blue) and 'Form Editor'.

```
CREATE TABLE `dim_weather` (
  `WeatherSK` int DEFAULT NULL,
  `WeatherCondition` varchar(100) DEFAULT NULL,
  `DI_CreatedDate` datetime DEFAULT NULL,
  `DI_WorkflowFileName` varchar(20) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

- Source Dimension



select * from finalProject_DataWarehouse.Dim_Source;

Query 1

```

46
47      #Source Dimension
48 •   select * from finalProject_DataWarehouse.Dim_Source;
49 •   select COUNT(*) from finalProject_DataWarehouse.Dim_Source;
50
  
```

SourceSK	SourceName	DI_CreatedDate	DI_WorkflowFileName
1	Texas-data.austintexas.gov	2024-04-07 15:34:44	load_dim_Source
2	Chicago Police Department	2024-04-07 15:34:46	load_dim_Source
3	Police Department (NYPD)	2024-04-07 15:34:57	load_dim_Source
HULL	HULL	HULL	HULL

select COUNT(*) from finalProject_DataWarehouse.Dim_Source;

The screenshot shows the MySQL Workbench interface. The query editor tab is titled "Query 1". The SQL code is as follows:

```
46
47      #Source Dimension
48 •  select * from finalProject_DataWarehouse.Dim_Source;
49 •  select COUNT(*) from finalProject_DataWarehouse.Dim_Source;
50
```

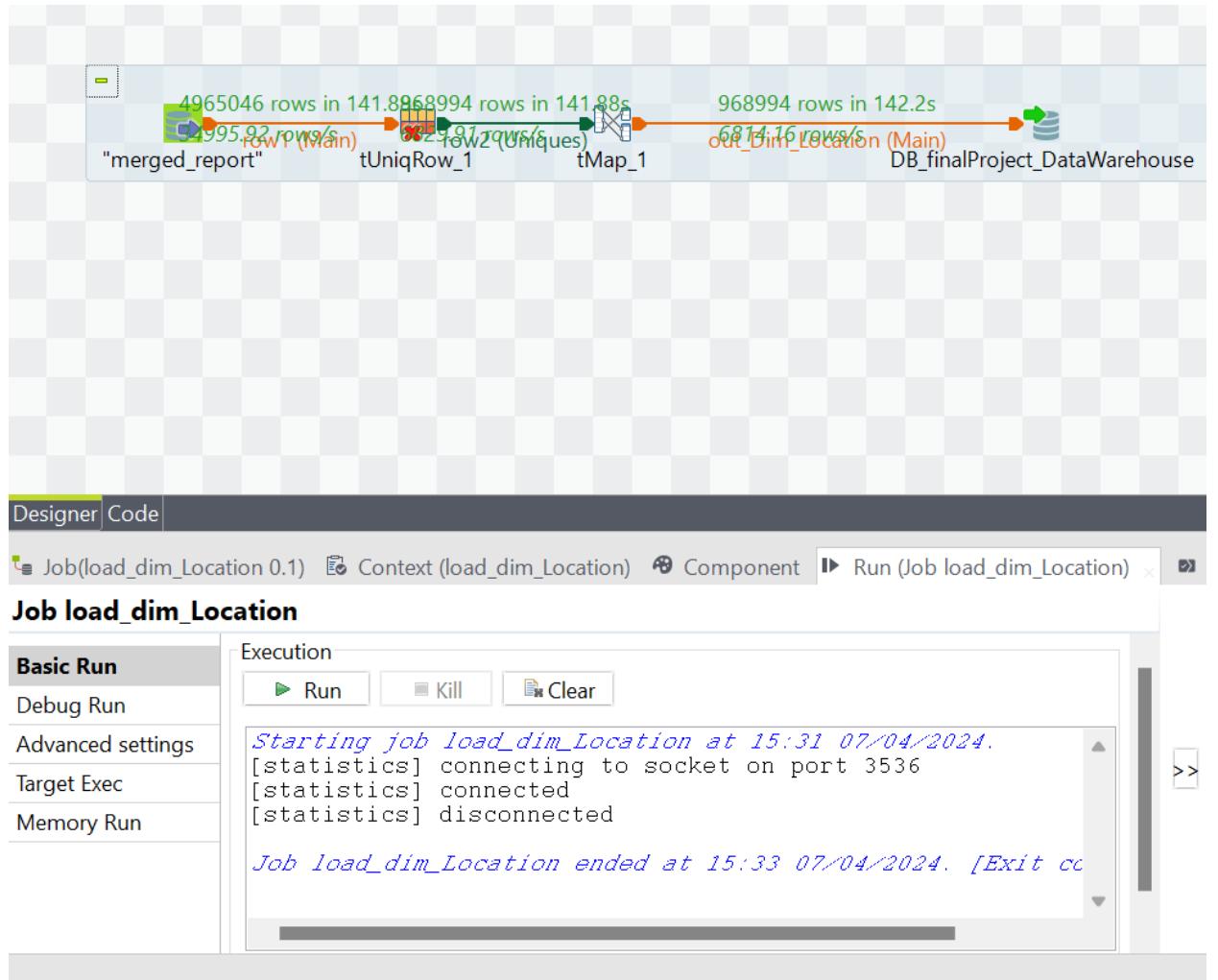
The result grid shows a single row with the value 3.

	COUNT(*)
▶	3

On the right side of the interface, there are two buttons: "Result Grid" (selected) and "Form Editor".

```
CREATE TABLE `dim_source` (
  `SourceSK` int NOT NULL,
  `SourceName` varchar(50) DEFAULT NULL,
  `DI_CreatedDate` datetime DEFAULT NULL,
  `DI_WorkflowFileName` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`SourceSK`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci
```

- Location Dimension



select * from finalProject_DataWarehouse.Dim_Location;

Query 1 SQL File 8 finaldamgproject.stg_contributin... finaldamgproject.transform_aust... finalproject_datawa...

```

50
51      #Location Dimension
52 •  select * from finalProject_DataWarehouse.Dim_Location;
53 •  select COUNT(*) from finalProject_DataWarehouse.Dim_Location;
54

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Result Grid | Form Editor | Field Types

	LocationSK	StreetAddress	Area	City	ZipCode	Latitude	Longitude
1	3707 MANCHACA RD	Nil	Austin	Nil			
2	PALM WAY TO MOPAC NB RAMP	Nil	Austin	Nil	30.404	-97.72445	
3	8849 BALCONES CLUB DR.	Nil	Austin	Nil	30.43798421	-97.78855804	
4	US0290	Nil	Austin	Nil	30.3276207970184	-97.6626808595204	
5	US0290	Nil	Austin	Nil	30.22515795	-97.76719944	
6	IH0035	Nil	Austin	Nil	30.16984339	-97.78425218	
7	4112 FM0973	Nil	Austin	Nil	30.19320516	-97.64740225	
8	1500 E ANDERSON LN	Nil	Austin	Nil	30.33279476	-97.68612246	
9	IH0035	Nil	Austin	Nil	30.29613303	-97.71883142	
10	SL0001	Nil	Austin	Nil	30.2657795	-97.78253937	
11	IH0035	Nil	Austin	Nil	30.2785745242044	-97.7304912979312	

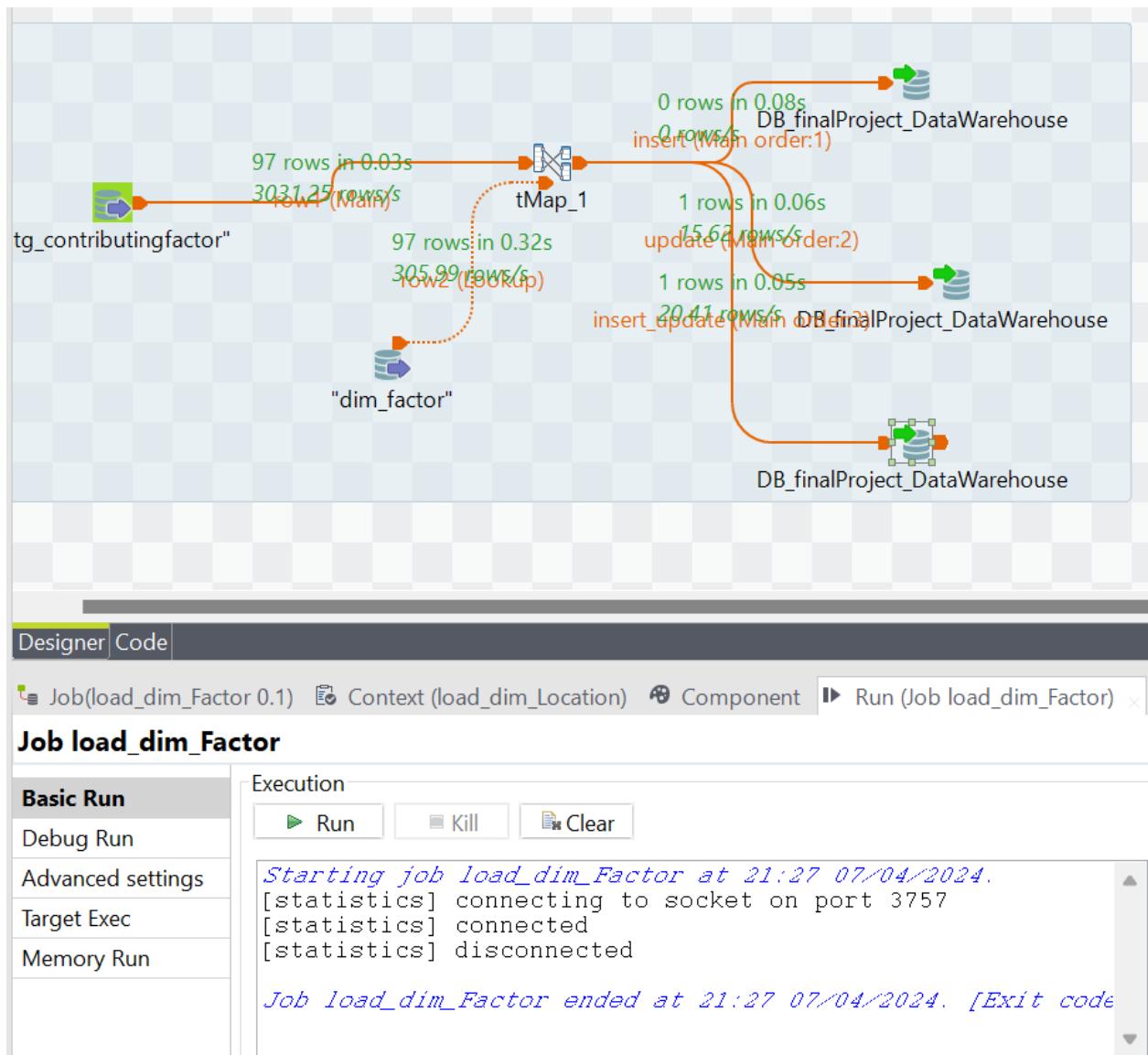
select COUNT(*) from finalProject_DataWarehouse.Dim_Location;

```
50
51      #Location Dimension
52 •   select * from finalProject_DataWarehouse.Dim_Location;
53 •   select COUNT(*) from finalProject_DataWarehouse.Dim_Location;
54
```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:	Result Grid	Form Editor
	COUNT(*)				968994	

```
CREATE TABLE `dim_location` (
  `LocationSK` int NOT NULL,
  `StreetAddress` varchar(500) DEFAULT NULL,
  `Area` varchar(20) DEFAULT NULL,
  `City` varchar(20) DEFAULT NULL,
  `ZipCode` varchar(10) DEFAULT NULL,
  `Latitude` varchar(20) DEFAULT NULL,
  `Longitude` varchar(20) DEFAULT NULL,
  `DI_CreatedDate` datetime DEFAULT NULL,
  `DI_WorkflowFileName` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`LocationSK`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci
```

- Factor Dimension



select * from finalProject_DataWarehouse.Dim_Factor;

```

55      #Factor Dimension
56 •   select * from finalProject_DataWarehouse.Dim_Factor;
57 •   select COUNT(*) from finalProject_DataWarehouse.Dim_Factor;
58
59      #Factor VehicleType
60 •   select * from finalProject_DataWarehouse.Dim_VehicleType;
61 •   select COUNT(*) from finalProject_DataWarehouse.Dim_VehicleType;

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

The screenshot shows a table titled 'Result Grid' with the following columns: FactorSK, FactorCode, FactorDescription, StartDate, EndDate, and ValidityIndicator. The data rows are as follows:

FactorSK	FactorCode	FactorDescription	StartDate	EndDate	ValidityIndicator
93	130	Pavement Defective	2024-04-07	NULL	Y
94	131	WEATHER	2024-04-07	NULL	Y
95	132	UNABLE TO DETERMINE	2024-04-07	NULL	Y
96	133	Driver Inexperience	2024-04-07	NULL	Y
97	134	Runaway Vehicle1	2024-04-07	2024-0...	N

```
select COUNT(*) from finalProject_DataWarehouse.Dim_Factor;
```

```

57 •   select COUNT(*) from finalProject_DataWarehouse.Dim_Factor;
58
59      #Factor VehicleType
60 •   select * from finalProject_DataWarehouse.Dim_VehicleType;
61 •   select COUNT(*) from finalProject_DataWarehouse.Dim_VehicleTyp

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

The screenshot shows a table titled 'Result Grid' with one column labeled 'COUNT(*)'. The value is 98.

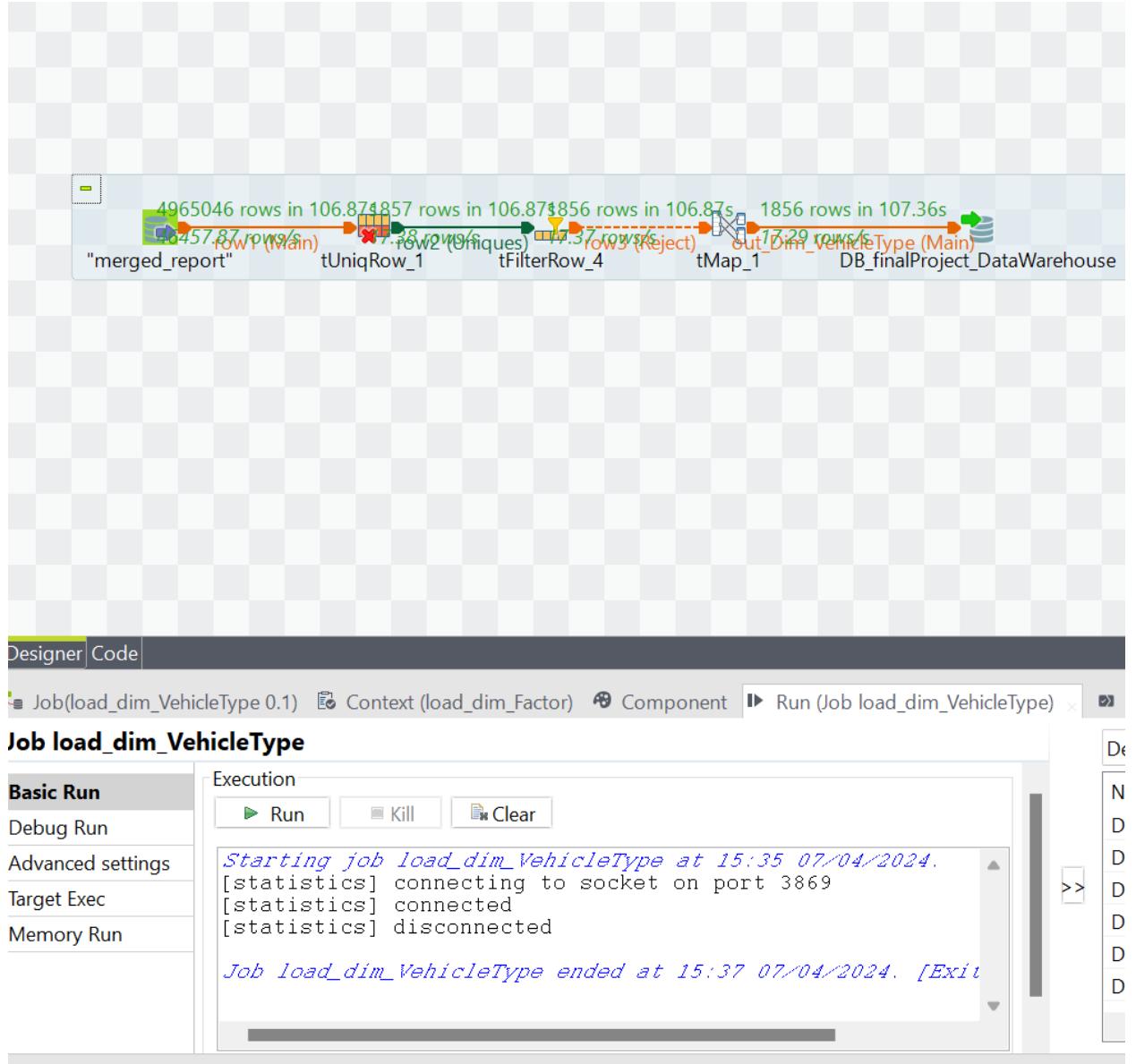
COUNT(*)
98

```

CREATE TABLE `dim_factor` (
  `FactorSK` int NOT NULL AUTO_INCREMENT,
  `FactorCode` int NOT NULL,
  `FactorDescription` varchar(1000) NOT NULL,
  `StartDate` date DEFAULT NULL,
  `EndDate` date DEFAULT NULL,
  `ValidityIndicator` varchar(2) DEFAULT NULL,
  PRIMARY KEY (`FactorSK`)
) ENGINE=InnoDB AUTO_INCREMENT=99 DEFAULT
CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci

```

- Dimension VehicleType



`select * from finalProject_DataWarehouse.Dim_VehicleType;`

Query 1 × SQL File 8 finaldamgproject.stg_contributin... finaldamgproject.transform_aust... finalproject_datawa...

```

58
59      #Factor VehicleType
60 •   select * from finalProject_DataWarehouse.Dim_VehicleType;
61 •   select * from finalProject_DataWarehouse.Dim_VehicleType;
62

```

Result Grid | Filter Rows: Edit: Export/Import: Wrap Cell Content: Result Grid Form Editor

	VehicleTypeSK	VehicleType	DI_CreatedDate	DI_WorkflowFileName
▶	1	Passenger car	2024-04-07 15:36:29	load_dim_VehicleType
	2	Large passenger vehicle	2024-04-07 15:36:29	load_dim_VehicleType
	3	Motor vehicle – other	2024-04-07 15:36:29	load_dim_VehicleType
	4	Motorcycle	2024-04-07 15:36:29	load_dim_VehicleType
		Other	2024-04-07 15:36:29	load_dim_VehicleType

select * from finalProject_DataWarehouse.Dim_VehicleType;

58
59 #Factor VehicleType
60 • select * from finalProject_DataWarehouse.Dim_VehicleType;
61 • select COUNT(*) from finalProject_DataWarehouse.Dim_VehicleType;
62

Result Grid | Filter Rows: Export: Wrap Cell Content: Result Grid Form Editor

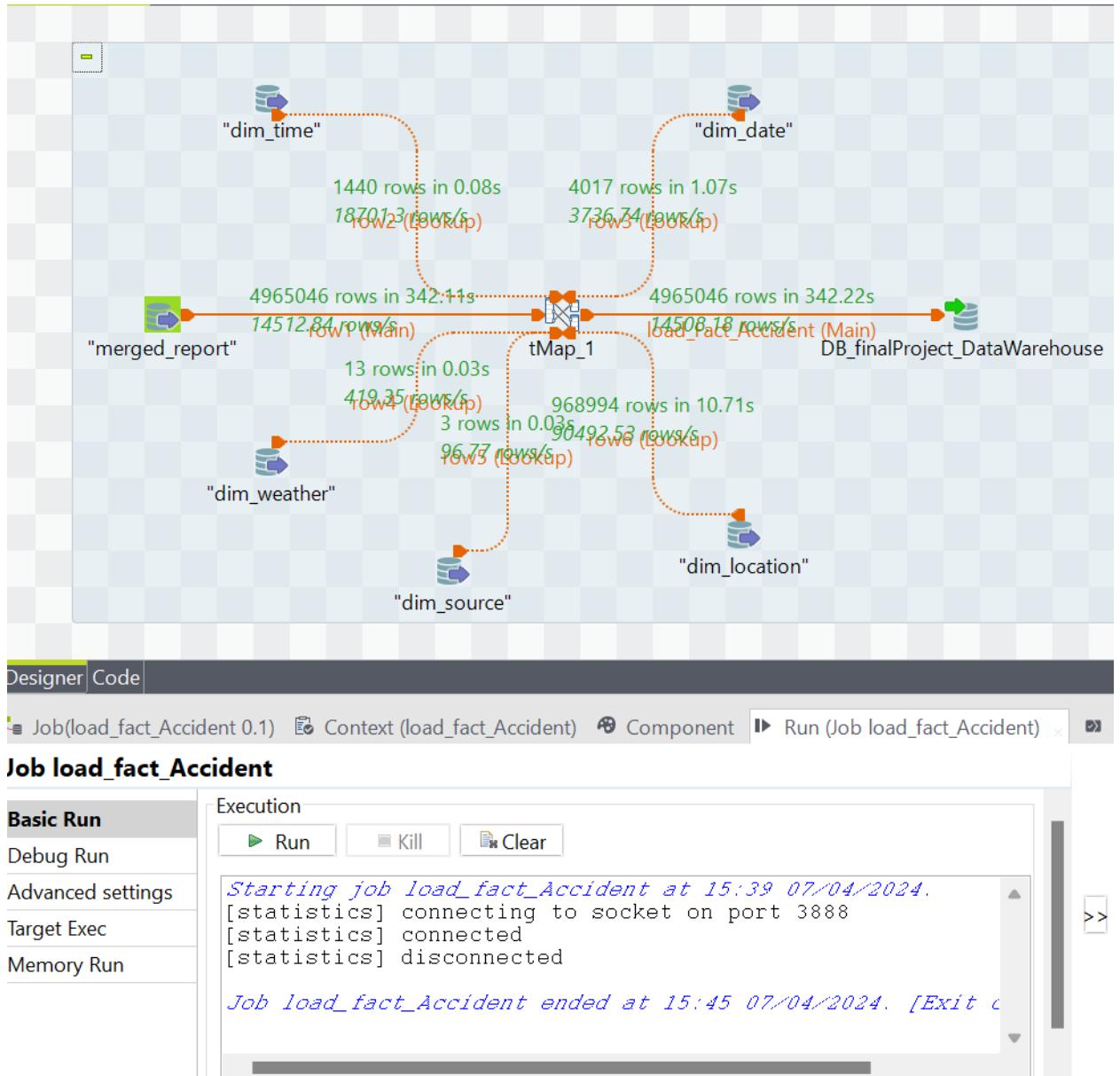
COUNT(*)
1856

```

CREATE TABLE `dim_vehicletype` (
  `VehicleTypeSK` int NOT NULL,
  `VehicleType` varchar(250) DEFAULT NULL,
  `DI_CreatedDate` datetime DEFAULT NULL,
  `DI_WorkflowFileName` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`VehicleTypeSK`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci

```

- Fact Accident



select * from finalProject_DataWarehouse.Fact_Accident;

Query 1 × SQL File 8 finaldamgproject.stg_contributin... finaldamgproject.transform_aust... finalproject_datawa

```

62
63      #Fact Accident
64 •  select * from finalProject_DataWarehouse.Fact_Accident;
65 •  select COUNT(*) from finalProject_DataWarehouse.Fact_Accident;
66

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Result Grid | Form

	AccidentSK	VehicleCount	MotoristInjured	MotoristKilled	PedestriansInjured	PedestriansKilled	CyclistInjured	CyclistKilled
▶	1	1	0	0	0	0	0	0
	2	1	0	0	0	0	0	0
	3	1	0	0	0	0	0	0
	4	1	0	0	0	0	0	0

select COUNT(*) from finalProject_DataWarehouse.Fact_Accident;

Query 1 × SQL File 8 finaldamgproject.stg_contributin... finaldamgproject.transform_aust... finalproject_datawa

```

62
63      #Fact Accident
64 •  select * from finalProject_DataWarehouse.Fact_Accident;
65 •  select COUNT(*) from finalProject_DataWarehouse.Fact_Accident;
66

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Result Grid | Form

	COUNT(*)
▶	4965046

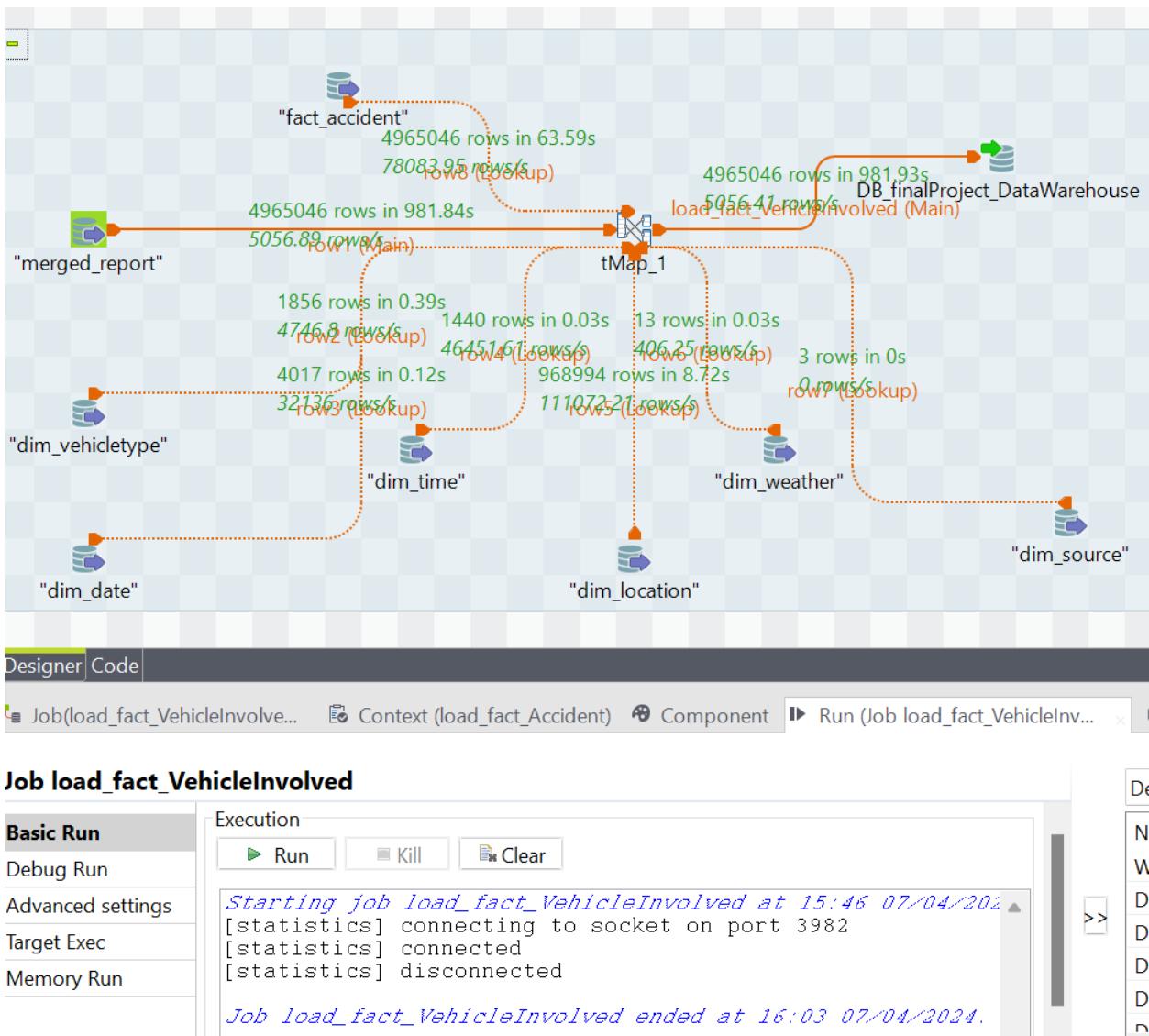
```

CREATE TABLE `fact_accident` (
  `AccidentSK` int NOT NULL,
  `VehicleCount` int DEFAULT NULL,
  `MotoristInjured` int DEFAULT NULL,
  `MotoristKilled` int DEFAULT NULL,
  `PedastriansInjured` int DEFAULT NULL,
  `PedastriansKilled` int DEFAULT NULL,
  `CyclistInjured` int DEFAULT NULL,
  `CyclistKilled` int DEFAULT NULL,
  `TotalInjured` int DEFAULT NULL,

```

```
'TotalKilled` int DEFAULT NULL,  
`DateSK` int NOT NULL,  
`TimeSK` varchar(15) NOT NULL,  
`LocationSK` int NOT NULL,  
`WeatherSK` int NOT NULL,  
`SourceSK` int NOT NULL,  
`DI_CreatedDate` datetime DEFAULT NULL,  
`DI_WorkflowFileName` varchar(50) DEFAULT NULL,  
PRIMARY KEY  
(`AccidentSK`, `DateSK`, `TimeSK`, `LocationSK`, `WeatherSK`, `SourceSK`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4  
COLLATE=utf8mb4_0900_ai_ci
```

- Fact VehicleInvolved



```
select * from finalProject_DataWarehouse.Fact_VehicleInvolved;
```

Query 1 × SQL File 8 finaldamgproject.stg_contributin... finaldamgproject.transform_aust... finalproject_datawa...

66
67 #Fact_VehicleInvolved
68 • select * from finalProject_DataWarehouse.Fact_VehicleInvolved;
69 • select COUNT(*) from finalProject_DataWarehouse.Fact_VehicleInvolved;
70

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Result Grid | Form Editor | Field Types | Query Stats

	VehicleInvolvedSK	VehicleTypeSK	AccidentSK	DI_CreatedDate	DI_WorkflowFileName
▶	1	1	1	2024-04-07 15:48:10	load_fact_VehicleInvolved
	2	2	2	2024-04-07 15:48:10	load_fact_VehicleInvolved
	3	2	3	2024-04-07 15:48:10	load_fact_VehicleInvolved
	4	3	4	2024-04-07 15:48:10	load_fact_VehicleInvolved
	5	1	5	2024-04-07 15:48:10	load_fact_VehicleInvolved
	6	1	6	2024-04-07 15:48:10	load_fact_VehicleInvolved
	7	1	7	2024-04-07 15:48:10	load_fact_VehicleInvolved
	8	2	8	2024-04-07 15:48:10	load_fact_VehicleInvolved
	9	1	9	2024-04-07 15:48:10	load_fact_VehicleInvolved
	10	1	10	2024-04-07 15:48:10	load_fact_VehicleInvolved
	11	2	11	2024-04-07 15:48:10	load_fact_VehicleInvolved
	12	2	13	2024-04-07 15:48:10	load_fact_VehicleInvolved
	13	1	13	2024-04-07 15:48:10	load_fact_VehicleInvolved
	14	1	13	2024-04-07 15:48:10	load_fact_VehicleInvolved

select COUNT(*) from finalProject_DataWarehouse.Fact_VehicleInvolved;

67 #Fact_VehicleInvolved
68 • select * from finalProject_DataWarehouse.Fact_VehicleInvolved;
69 • select COUNT(*) from finalProject_DataWarehouse.Fact_VehicleInvolved;
70

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Result Grid | Form Editor | Field Types | Query Stats

COUNT(*)
4965046

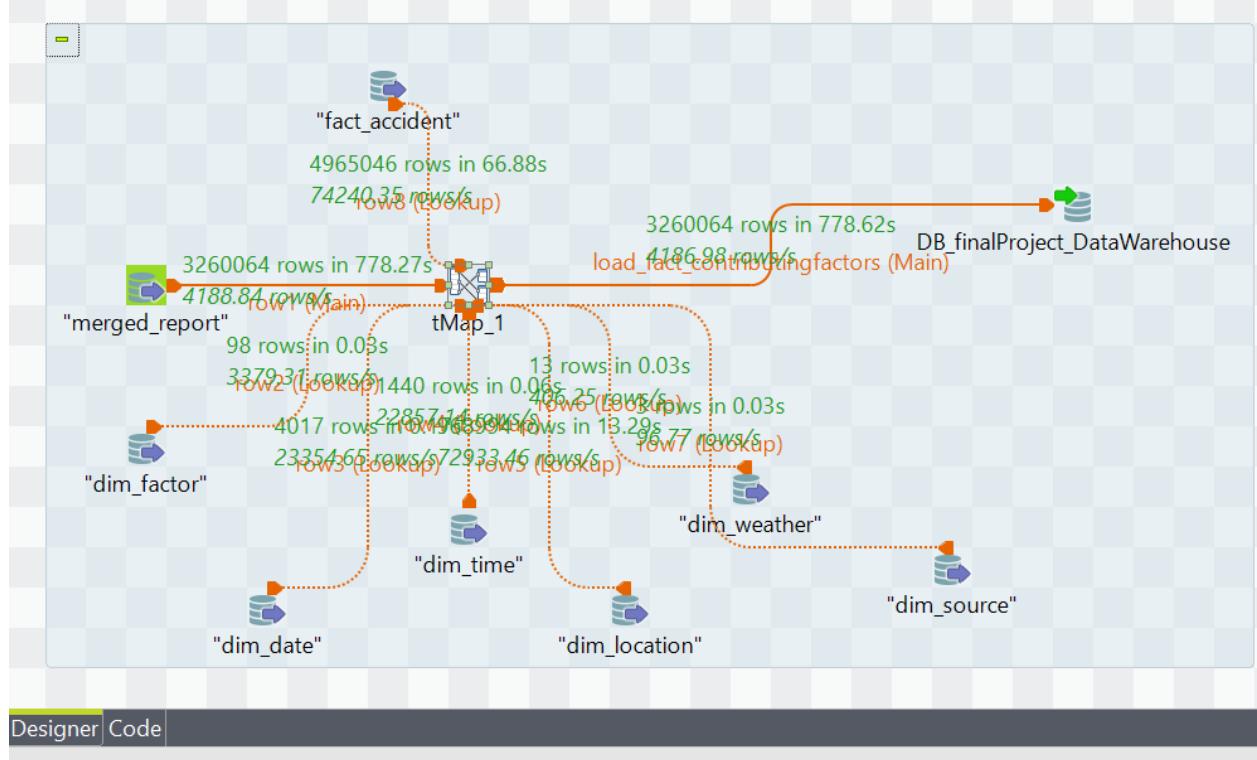
CREATE TABLE `fact_vehicleinvolved` (
`VehicleInvolvedSK` int NOT NULL,
`VehicleTypeSK` int NOT NULL,
`AccidentSK` int NOT NULL,
`DI_CreatedDate` datetime DEFAULT NULL,
`DI_WorkflowFileName` varchar(50) DEFAULT NULL,

```

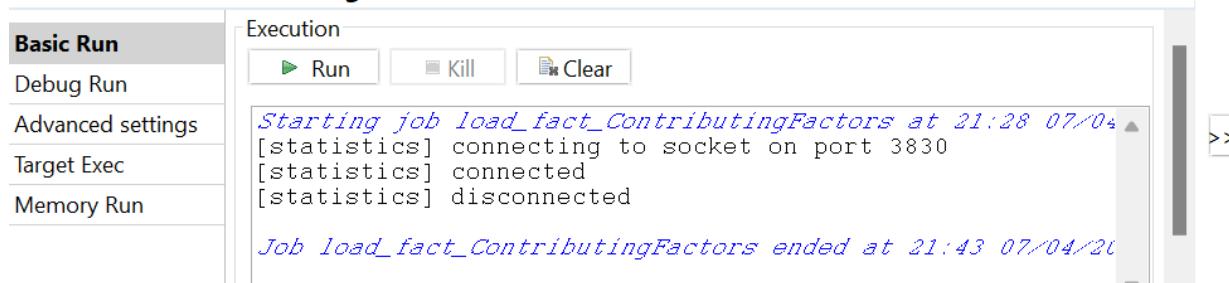
PRIMARY KEY (`VehicleInvolvedSK`, `VehicleTypeSK`, `AccidentSK`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci

```

- Fact ContributingFactors



Job load_fact_ContributingFactors



```
select * from finalProject_DataWarehouse.Fact_ContributingFactors;
```

Query 1 x SQL File 8 finaldamgproject.stg_contributin... finaldamgproject.transform_aust... finalproject_datawa

69 • select COUNT(*) from finalProject_DataWarehouse.Fact_VehicleInvolved;
 70
 71 #Fact_ContributingFactors
 72 • select * from finalProject_DataWarehouse.Fact_ContributingFactors;
 73 • select COUNT(*) from finalProject_DataWarehouse.Fact_ContributingFactors;

	ContributingFactorSK	FactorSK	AccidentSK	DI_CreatedDate	DI_WorkflowFileName
▶	1	1	1	2024-04-07 16:08:54	load_fact_ContributingFactors
	2	1	2	2024-04-07 16:08:54	load_fact_ContributingFactors
	3	1	3	2024-04-07 16:08:54	load_fact_ContributingFactors
	4	1	4	2024-04-07 16:08:54	load_fact_ContributingFactors
	5	2	5	2024-04-07 16:08:54	load_fact_ContributingFactors
	6	1	6	2024-04-07 16:08:54	load_fact_ContributingFactors
	7	1	7	2024-04-07 16:08:54	load_fact_ContributingFactors
	8	1	8	2024-04-07 16:08:54	load_fact_ContributingFactors
	9	1	9	2024-04-07 16:08:54	load_fact_ContributingFactors
	10	1	10	2024-04-07 16:08:54	load_fact_ContributingFactors
	11	3	11	2024-04-07 16:08:54	load fact ContributingFactors

select COUNT(*) from finalProject_DataWarehouse.Fact_ContributingFactors;

Query 1 x SQL File 8 finaldamgproject.stg_contributin... finaldamgproject.transform_aust... finalproject_datawa

69 • select COUNT(*) from finalProject_DataWarehouse.Fact_VehicleInvolved;
 70
 71 #Fact_ContributingFactors
 72 • select * from finalProject_DataWarehouse.Fact_ContributingFactors;
 73 • select COUNT(*) from finalProject_DataWarehouse.Fact_ContributingFactors;

	COUNT(*)
▶	4965046

```
CREATE TABLE `fact_contributingfactors` (
  `ContributingFactorSK` int NOT NULL,
  `FactorSK` int NOT NULL,
  `AccidentSK` int NOT NULL,
  `DI_CreatedDate` datetime DEFAULT NULL,
  `DI_WorkflowFileName` varchar(50) DEFAULT NULL,
```

```
PRIMARY KEY(`ContributingFactorSK`, `FactorSK`, `AccidentSK`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

SQL Scripts to Validate Business Questions

1. How many accidents occurred in NYC, Austin, and Chicago?

```
SELECT
    dl.City,
    COUNT(*) AS AccidentCount
FROM fact_accident fa
JOIN dim_location dl ON fa.LocationSK = dl.LocationSK
WHERE dl.City IN ('New York', 'Austin', 'Chicago')
GROUP BY dl.City;
```

Query 1 × SQL File 8 finaldamgproject.stg_contributin... finaldamgproject.transform_aust...

61
62 #Fact_ContributingFactors
63 • select * from finalProject_DataWarehouse.Fact_ContributingFactors;
64
65 • use `finalproject_datawarehouse`;
66
67 #How many accidents occurred in NYC, Austin, and Chicago?
68 • SELECT
69 dl.City,
70 COUNT(*) AS AccidentCount
71 FROM fact_accident fa
72 JOIN dim_location dl ON fa.LocationSK = dl.LocationSK
73 WHERE dl.City IN ('New York', 'Austin', 'Chicago')
74 GROUP BY dl.City;
75
76 #Which areas in the 3 cities had the greatest number of accidents? (top 3 areas in each
77 • SELECT

Result Grid | Filter Rows: Export: Wrap Cell Content: Result Grid

	City	AccidentCount
▶	Austin	228259
	Chicago	1350460
	New York	3382116

2. Which areas in the 3 cities had the greatest number of accidents? (Top 3 areas in each city)

```
SELECT
    dl.City,
    dl.Area,
    COUNT(*) AS AccidentCount
FROM fact_accident fa
JOIN dim_location dl ON fa.LocationSK = dl.LocationSK
WHERE dl.City IN ('New York', 'Austin', 'Chicago')
    AND NOT (dl.City = 'New York' AND (dl.Area IS NULL OR dl.Area = ""))
GROUP BY dl.City, dl.Area
ORDER BY dl.City, AccidentCount DESC
LIMIT 3;
```

Query 1 x SQL File 8 finaldamgproject.stg_contributin... finaldamgproject.transform_aust...

76 #Which areas in the 3 cities had the greatest number of accidents? (top 3 areas in each
77 • SELECT
78 dl.City,
79 dl.Area,
80 COUNT(*) AS AccidentCount
81 FROM fact_accident fa
82 JOIN dim_location dl ON fa.LocationSK = dl.LocationSK
83 WHERE dl.City IN ('New York', 'Austin', 'Chicago')
84 AND NOT (dl.City = 'New York' AND (dl.Area IS NULL OR dl.Area = ''))
85 GROUP BY dl.City, dl.Area
86 ORDER BY dl.City, AccidentCount DESC
87 LIMIT 3;
88
89 # How many accidents resulted in just injuries? (Overall and by city)
90 -- Overall
91 • SELECT
92 COUNT(*) AS AccidentsWithInjuries

Result Grid | Filter Rows: Export: Wrap Cell Content: Fetch rows: Result Grid

	City	Area	AccidentCount
▶	Austin	Nil	228259
	Chicago	Nil	1350460
	New York	BROOKLYN	721048

3. How many accidents resulted in just injuries? (Overall and by city)

```
-- Overall
SELECT
  COUNT(*) AS AccidentsWithInjuries
FROM fact_accident
WHERE TotalInjured > 0 AND TotalKilled = 0;
```

```
-- By City
SELECT
  dl.City,
  COUNT(*) AS AccidentsWithInjuries
FROM fact_accident fa
```

```
JOIN dim_location dl ON fa.LocationSK = dl.LocationSK  
WHERE TotalInjured > 0 AND TotalKilled = 0  
GROUP BY dl.City;
```

Query 1 x SQL File 8 finaldamgproject.stg_contributin... finaldamgproject.transform_aust...

The screenshot shows a SQL query editor window with the following details:

- Toolbar:** Includes icons for file operations, search, and navigation.
- Query Editor:** Displays the following SQL code:

```
89    # How many accidents resulted in just injuries? (Overall and by city)  
90    -- Overall  
91 •     SELECT  
92         COUNT(*) AS AccidentsWithInjuries  
93     FROM fact_accident  
94     WHERE TotalInjured > 0 AND TotalKilled = 0;  
95  
96  
97    -- By City  
98 •     SELECT  
99        dl.City,  
100       COUNT(*) AS AccidentsWithInjuries  
101      FROM fact_accident fa  
102      JOIN dim_location dl ON fa.LocationSK = dl.LocationSK  
103      WHERE TotalInjured > 0 AND TotalKilled = 0  
104      GROUP BY dl.City;  
105
```
- Result Grid:** Shows the output of the query:

AccidentsWithInjuries
1040948
- Buttons:** Includes "Result Grid" (highlighted in blue), "Filter Rows:", "Export:", and "Wrap Cell Content:".

Query 1 SQL File 8 finaldamgproject.stg_contributin... finaldamgproject.transform_aust...

89 # How many accidents resulted in just injuries? (Overall and by city)
90 -- Overall
91 • SELECT
92 COUNT(*) AS AccidentsWithInjuries
93 FROM fact_accident
94 WHERE TotalInjured > 0 AND TotalKilled = 0;
95
96
97 -- By City
98 • SELECT
99 dl.City,
100 COUNT(*) AS AccidentsWithInjuries
101 FROM fact_accident fa
102 JOIN dim_location dl ON fa.LocationSK = dl.LocationSK
103 WHERE TotalInjured > 0 AND TotalKilled = 0
104 GROUP BY dl.City;
105

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Result Grid

	City	AccidentsWithInjuries
▶	Austin	103396
	Chicago	193115
	New York	743463

4. How often are pedestrians involved in accidents? (Overall and by city)

```
-- Overall
SELECT
    COUNT(*) AS PedestrianInvolvedAccidents
FROM fact_accident
WHERE PedastriansInjured > 0 OR PedastriansKilled > 0;

-- By City
SELECT
    dl.City,
    COUNT(*) AS PedestrianInvolvedAccidents
FROM fact_accident fa
JOIN dim_location dl ON fa.LocationSK = dl.LocationSK
WHERE PedastriansInjured > 0 OR PedastriansKilled > 0
GROUP BY dl.City;
```

Query 1 × SQL File 8 finaldamgproject.stg_contributin... finaldamgproject.transform_aust...

104 GROUP BY dl.City;

105

106 # How often are pedestrians involved in accidents? (Overall and by city)

107 -- Overall

108 • SELECT

109 COUNT(*) AS PedestrianInvolvedAccidents

110 FROM fact_accident

111 WHERE PedastriansInjured > 0 OR PedastriansKilled > 0;

112

113 -- By City

114 • SELECT

115 dl.City,

116 COUNT(*) AS PedestrianInvolvedAccidents

117 FROM fact_accident fa

118 JOIN dim_location dl ON fa.LocationSK = dl.LocationSK

119 WHERE PedastriansInjured > 0 OR PedastriansKilled > 0

120 GROUP BY dl.City;

Result Grid | Filter Rows: Export: Wrap Cell Content:

	PedestrianInvolvedAccidents
▶	119320

Result Grid

Query 1 × SQL File 8 finaldamgproject.stg_contributin... finaldamgproject.transform_aust...

110 FROM fact_accident
111 WHERE PedastriansInjured > 0 OR PedastriansKilled > 0;
112
113 -- By City
114 • SELECT
115 dl.City,
116 COUNT(*) AS PedestrianInvolvedAccidents
117 FROM fact_accident fa
118 JOIN dim_location dl ON fa.LocationSK = dl.LocationSK
119 WHERE PedastriansInjured > 0 OR PedastriansKilled > 0
120 GROUP BY dl.City;
121
122 # When do most accidents happen? (Seasonality report)
123 • SELECT
124 dd.Quarter,
125 dd.MonthName,
126 COUNT(*) AS AccidentCount

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Result Grid

	City	PedestrianInvolvedAccidents
▶	Austin	2037
	New York	117193

5. When do most accidents happen? (Seasonality report)

```
SELECT
  dd.Quarter,
  dd.MonthName,
  COUNT(*) AS AccidentCount
FROM fact_accident fa
JOIN dim_date dd ON fa.DateSK = dd.DateSK
GROUP BY dd.Quarter, dd.MonthName
ORDER BY dd.Quarter, MIN(dd.MonthNumberOfYear)
LIMIT 0, 1000;
```

Query 1 > SQL File 8 finaldamgproject.stg_contributin... finaldamgproject.transform_aust...

116 COUNT(*) AS PedestrianInvolvedAccidents
117 FROM fact_accident fa
118 JOIN dim_location dl ON fa.LocationSK = dl.LocationSK
119 WHERE PedestriansInjured > 0 OR PedestriansKilled > 0
120 GROUP BY dl.City;
121
122 # When do most accidents happen? (Seasonality report)
123 • SELECT
124 dd.Quarter,
125 dd.MonthName,
126 COUNT(*) AS AccidentCount
127 FROM fact_accident fa
128 JOIN dim_date dd ON fa.DateSK = dd.DateSK
129 GROUP BY dd.Quarter, dd.MonthName
130 ORDER BY dd.Quarter, MIN(dd.MonthNumberOfYear)
131 LIMIT 0, 1000;
132

Result Grid | Filter Rows: Export: Wrap Cell Content: Result Grid Read Only

	Quarter	MonthName	AccidentCount
▶	1	January	366261
	1	February	346647
	1	March	378388
	2	April	358099
	2	May	412982

Result 47 x

Result Grid | Filter Rows: Export:

	Quarter	MonthName	AccidentCount
▶	1	January	366261
	1	February	346647
	1	March	378388
	2	April	358099
	2	May	412982
	2	June	416479
	3	July	408408
	3	August	407148
	3	September	411473
	4	October	428588
	4	November	397602
	4	December	398222

6. How many motorists are injured or killed in accidents? (Overall and by city)

```
-- Overall
SELECT
    SUM(MotoristInjured) AS TotalMotoristInjured,
    SUM(MotoristKilled) AS TotalMotoristKilled
FROM fact_accident;
```

```
-- By City
SELECT
    dl.City,
    SUM(MotoristInjured) AS TotalMotoristInjured,
    SUM(MotoristKilled) AS TotalMotoristKilled
FROM fact_accident fa
JOIN dim_location dl ON fa.LocationSK = dl.LocationSK
GROUP BY dl.City;
```

The screenshot shows a SQL query editor interface with the following details:

- Query 1**: The current query tab.
- SQL File 8**: The file containing the queries.
- finaldamgproject.stg_contributin...**: The schema or table being queried.
- finaldamgproject.transform_aust...**: Another schema or table being queried.
- Toolbar icons**: Standard database and query execution icons.
- Line numbers**: Numbered from 131 to 147, corresponding to the SQL code lines.
- Code content**: The two SQL scripts listed above.
- Result Grid**: A table showing the results of the second query.

	TotalMotoristInjured	TotalMotoristKilled
▶	807357	2916

- Buttons**: Filter Rows, Export, Wrap Cell Content, and Result Grid button.

Query 1 x SQL File 8 finaldamgproject.stg_contributin ... finaldamgproject.transform_aust...

134 -- Overall
135 • SELECT
136 SUM(MotoristInjured) AS TotalMotoristInjured,
137 SUM(MotoristKilled) AS TotalMotoristKilled
138 FROM fact_accident;
139
140 -- By City
141 • SELECT
142 dl.City,
143 SUM(MotoristInjured) AS TotalMotoristInjured,
144 SUM(MotoristKilled) AS TotalMotoristKilled
145 FROM fact_accident fa
146 JOIN dim_location dl ON fa.LocationSK = dl.LocationSK
147 GROUP BY dl.City;
148
149 # Which top 5 areas in 3 cities have the most fatal number of accidents?
150 • SELECT

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Result Grid |

	City	TotalMotoristInjured	TotalMotoristKilled
▶	Austin	5236	680
	Chicago	NULL	NULL
	New York	801004	2226

7. Which top 5 areas in 3 cities have the most fatal number of accidents?

```
SELECT
    dl.City,
    dl.Area,
    SUM(TotalKilled) AS FatalAccidents
FROM fact_accident fa
JOIN dim_location dl ON fa.LocationSK = dl.LocationSK
GROUP BY dl.City, dl.Area
ORDER BY dl.City, FatalAccidents DESC
LIMIT 5;
```

Query 1 x SQL File 8 finaldamgproject.stg_contributin... finaldamgproject.transform_aust...

```

146     JOIN dim_location dl ON fa.LocationSK = dl.LocationSK
147     GROUP BY dl.City;
148
149     # Which top 5 areas in 3 cities have the most fatal number of accidents?
150 •   SELECT
151     dl.City,
152     dl.Area,
153     SUM(TotalKilled) AS FatalAccidents
154     FROM fact_accident fa
155     JOIN dim_location dl ON fa.LocationSK = dl.LocationSK
156     GROUP BY dl.City, dl.Area
157     ORDER BY dl.City, FatalAccidents DESC
158     LIMIT 5;
159
160     # Time-based analysis of accidents (Time of the day, day of the week, weekdays or weeke
161     -- Time of the Day
162 •   SELECT

```

Result Grid | Filter Rows: Export: Wrap Cell Content: Fetch rows: Result Grid Read Only

	City	Area	FatalAccidents
▶	Austin	Nil	1683
	Chicago	Nil	1611
	New York		1807
	New York	BROOKLYN	942
	New York	QUEENS	746

Result 50 x

8. Time-based analysis of accidents (Time of the day, day of the week, weekdays or weekends)

```
-- Time of the Day
SELECT
dt.TimeOfDay,
COUNT(*) AS AccidentCount
FROM fact_accident fa
JOIN dim_time dt ON fa.TimeSK = dt.TimeSK
GROUP BY dt.TimeOfDay;

-- Day of the Week
SELECT
dd.DayNameOfWeek,
```

```

        COUNT(*) AS AccidentCount
FROM fact_accident fa
JOIN dim_date dd ON fa.DateSK = dd.DateSK
GROUP BY dd.DayNameOfWeek;

-- Weekdays vs Weekends
SELECT
CASE
    WHEN dd.WeekendIndicator = 'Yes' THEN 'Weekend'
    ELSE 'Weekday'
END AS DayType,
COUNT(*) AS AccidentCount
FROM fact_accident fa
JOIN dim_date dd ON fa.DateSK = dd.DateSK
GROUP BY dd.WeekendIndicator;

```

159
160 # Time-based analysis of accidents (Time of the day, day of the week, weekdays or weekend days)
161 -- Time of the Day
162 • SELECT
163 dt.TimeOfDay,
164 COUNT(*) AS AccidentCount
165 FROM fact_accident fa
166 JOIN dim_time dt ON fa.TimeSK = dt.TimeSK
167 GROUP BY dt.TimeOfDay;
168

TimeOfDay	AccidentCount
10:58:00	1638
13:07:00	1757
15:42:00	1692
14:09:00	1497
18:00:00	57913
03:26:00	373
14:34:00	1541
02:06:00	488
00:11:00	774
10:05:00	5829
13:06:00	1458
16:04:00	1038

```

168
169      -- Day of the Week
170 •   SELECT
171         dd.DayNameOfWeek,
172         COUNT(*) AS AccidentCount
173     FROM fact_accident fa
174     JOIN dim_date dd ON fa.DateSK = dd.DateSK
175     GROUP BY dd.DayNameOfWeek;
176
177      -- Weekdays vs Weekends

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

DayNameOfWeek	AccidentCount
Sunday	559067
Thursday	705153
Friday	765287
Wednesday	689591
Monday	668912
Tuesday	694456
Saturday	647831

Result Grid | Form Editor |

Query 1 x SQL File 8 finaldamgproject.stg_contributin... finaldamgproject.transform_aust...



```

176
177      -- Weekdays vs Weekends
178 •   SELECT
179         CASE
180             WHEN dd.WeekendIndicator = 'Yes' THEN 'Weekend'
181             ELSE 'Weekday'
182         END AS DayType,
183         COUNT(*) AS AccidentCount
184     FROM fact_accident fa
185     JOIN dim_date dd ON fa.DateSK = dd.DateSK
186     GROUP BY dd.WeekendIndicator;
187

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

DayType	AccidentCount
Weekend	1206898
Weekday	3523399

Result Grid |

9. Fatality analysis

```
-- Are pedestrians killed more often than road users?
```

```
SELECT
```

```
'Pedestrians' AS VictimType,  
SUM(PedastriansKilled) AS Deaths
```

```
FROM fact_accident
```

```
UNION ALL
```

```
SELECT
```

```
'Motorists' AS VictimType,  
SUM(MotoristKilled) AS Deaths
```

```
FROM fact_accident
```

```
UNION ALL
```

```
SELECT
```

```
'Cyclists' AS VictimType,  
SUM(CyclistKilled) AS Deaths
```

```
FROM fact_accident;
```

Query 1 SQL File 8 finaldamgproject.stg_contributin... finaldamgproject.transform_aust...

189 -- Are pedestrians killed more often than road users?

190 • SELECT
191 'Pedestrians' AS VictimType,
192 SUM(PedastriansKilled) AS Deaths
193 FROM fact_accident
194 UNION ALL
195 SELECT
196 'Motorists' AS VictimType,
197 SUM(MotoristKilled) AS Deaths
198 FROM fact_accident
199 UNION ALL
200 SELECT
201 'Cyclists' AS VictimType,
202 SUM(CyclistKilled) AS Deaths
203 FROM fact_accident;
204
205 # What are the most common factors involved in accidents?

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

VictimType	Deaths
Pedestrians	2492
Motorists	2916
Cyclists	753

Result Grid

10. What are the most common factors involved in accidents?

```
SELECT
    df.FactorDescription,
    COUNT(*) AS Occurrences
FROM fact_contributingfactors fcf
JOIN dim_factor df ON fcf.FactorSK = df.FactorSK
GROUP BY df.FactorDescription
ORDER BY Occurrences DESC
LIMIT 10;
```

Query 1 > SQL File 8 finaldamgproject.stg_contributin ... finaldamgproject.transform_aust...

205 # What are the most common factors involved in accidents?
206 • SELECT
207 df.FactorDescription,
208 COUNT(*) AS Occurrences
209 FROM fact_contributingfactors fcf
210 JOIN dim_factor df ON fcf.FactorSK = df.FactorSK
211 GROUP BY df.FactorDescription
212 ORDER BY Occurrences DESC
213 LIMIT 10;

Result Grid | Filter Rows: Export: Wrap Cell Content: Fetch rows:

FactorDescription	Occurrences
UNABLE TO DETERMINE	1656134
Driver Inattention	699106
other	338650
Reaction to Uninvolved Vehicle	291924
Followed Too Closely	287659
Failed to Drive in Single Lane	245614
Fatigued or Asleep	136827
Backed without Safety	126544
Unsafe Speed	116731
FAILING TO YIELD RIGHT-OF-WAY	104173

Result Grid | Form Editor | Field Types | Query Stats