

Northeastern University

Milestone 7

IE 7374 Data Warehousing and Business Intelligence

Spring 2024

Team Member's List

Meet Vasani

Mihir Kakadiya

Formula 1 Race Analysis

Problem Definition

Formula 1, spanning from 1950 to 2017, holds a rich dataset encompassing various aspects of racing. The challenge is to extract valuable insights and optimize strategies based on the provided tables: circuits, constructors, drivers, laptimes, pitstops, qualifying, races, results, seasons, and status.

Key Objectives:

Historical Performance Analysis: Explore the dataset to understand how circuits, constructors, and drivers have evolved over the years. Identify trends, successes, and challenges.

Driver and Constructor Evaluation: Assess the performance of individual drivers and constructors. Identify standout performers, analyze consistency, and determine factors contributing to success or struggle.

Lap Time Optimization: Analyze lap times to identify patterns and areas for improvement. Optimize car setups and strategies to enhance overall performance on different circuits.

Pit Stop Strategies: Evaluate pitstop data to refine strategies. Minimize pitstop times, identify optimal windows for pit stops, and mitigate potential issues.

Qualifying Performance: Understand historical qualifying data to analyze team and driver capabilities in securing favorable starting positions. Identify trends that correlate with race performance.

Race Outcomes and Points Distribution: Analyze race results and points earned to understand the impact of different factors on overall standings. Identify instances of strategic brilliance or missed opportunities.

Seasonal Trends and Performance: Examine data across seasons to identify overarching trends, challenges, and successes. Pinpoint seasons with exceptional team or driver performances.

Real-time Decision Support: Leverage real-time status data to simulate scenarios during races. Develop decision support mechanisms for teams to adapt strategies based on changing conditions.

Dataset Attributes

Circuits: Different circuit names with their location.

Car Constructor: Car constructor name and its location.

Drivers: Driver information.

Laptimes: Different laptimes as per different drivers.

Pitstops: Number of stops as per different race and drivers.

Qualifying: Top 3 qualifying drivers as per different race.

Races: Race details as per circuits.

Results: Results as per all other tables.

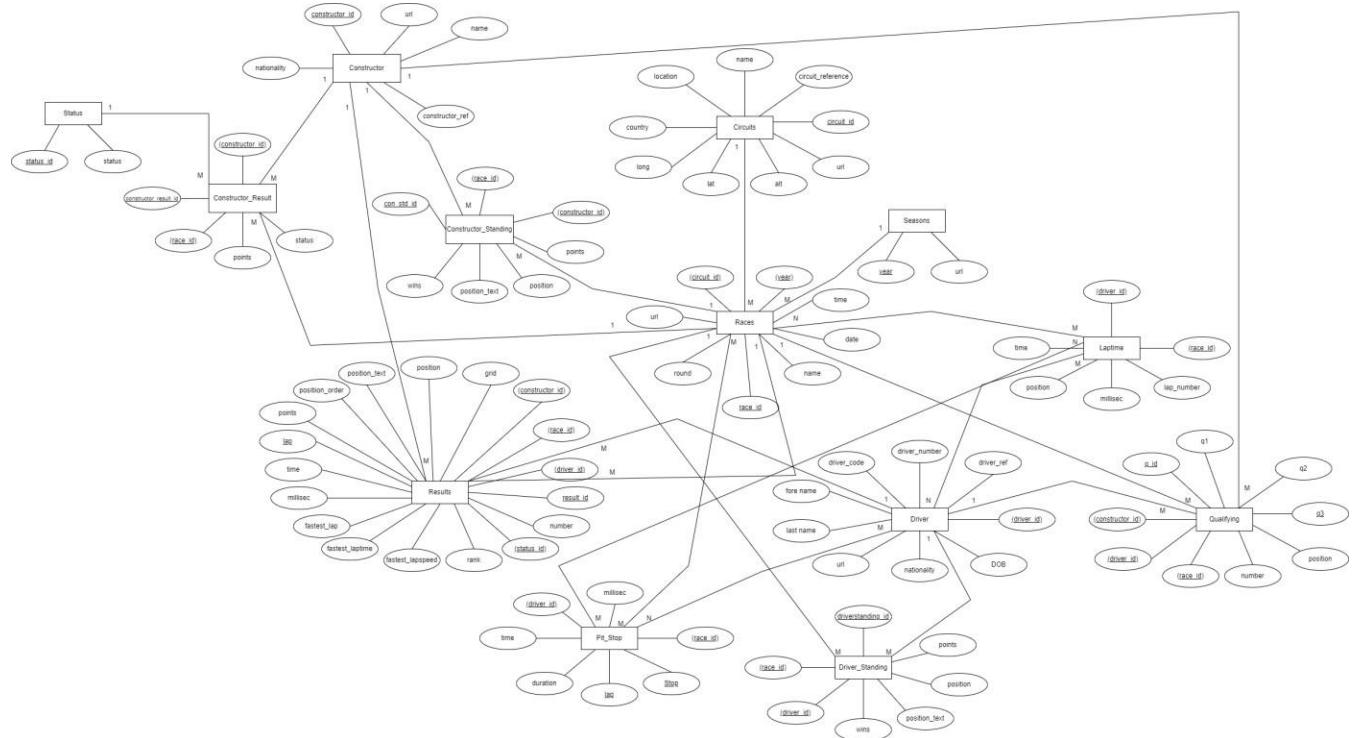
Seasons: A list of every season and corresponding Wikipedia link.

Status: A table of status codes and their status.

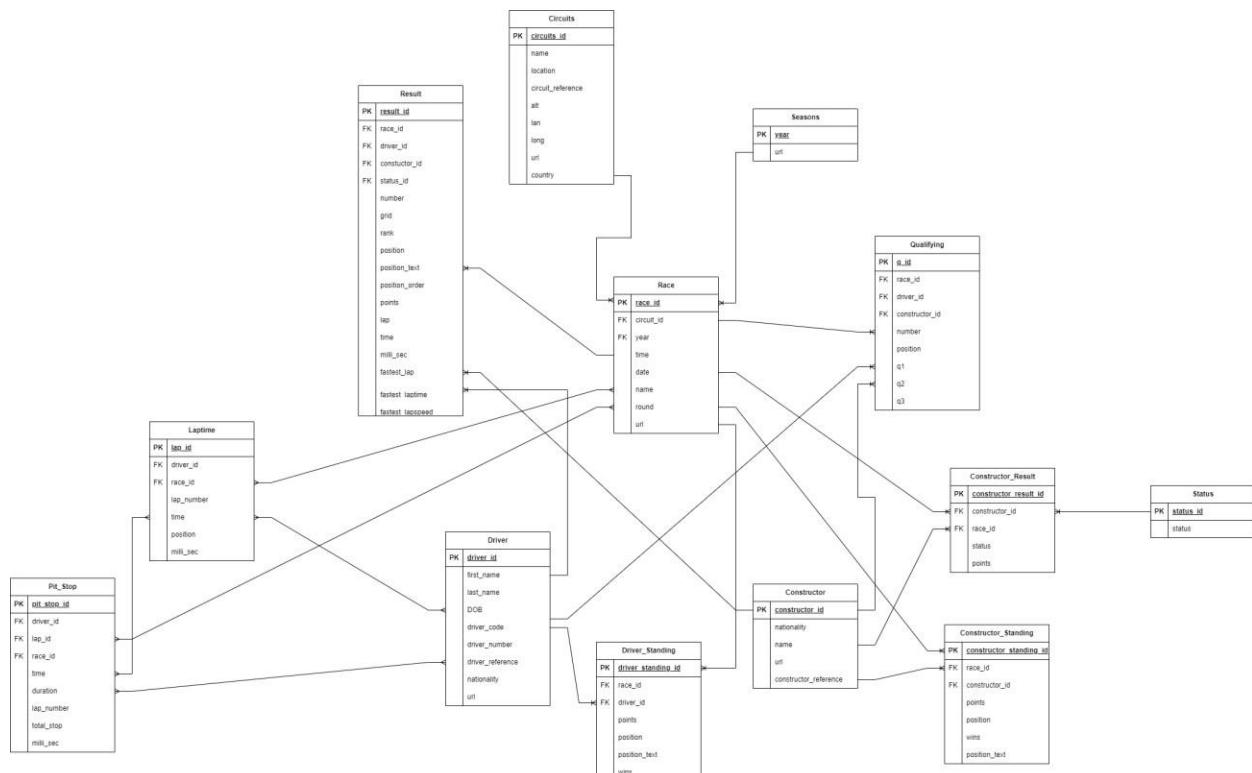
Expected Outputs

The project aims to provide teams, analysts, and enthusiasts with actionable insights derived from historical Formula 1 data. This includes improved decision-making strategies, optimized car setups, and a deeper understanding of the factors influencing success in one of the world's premier racing championships.

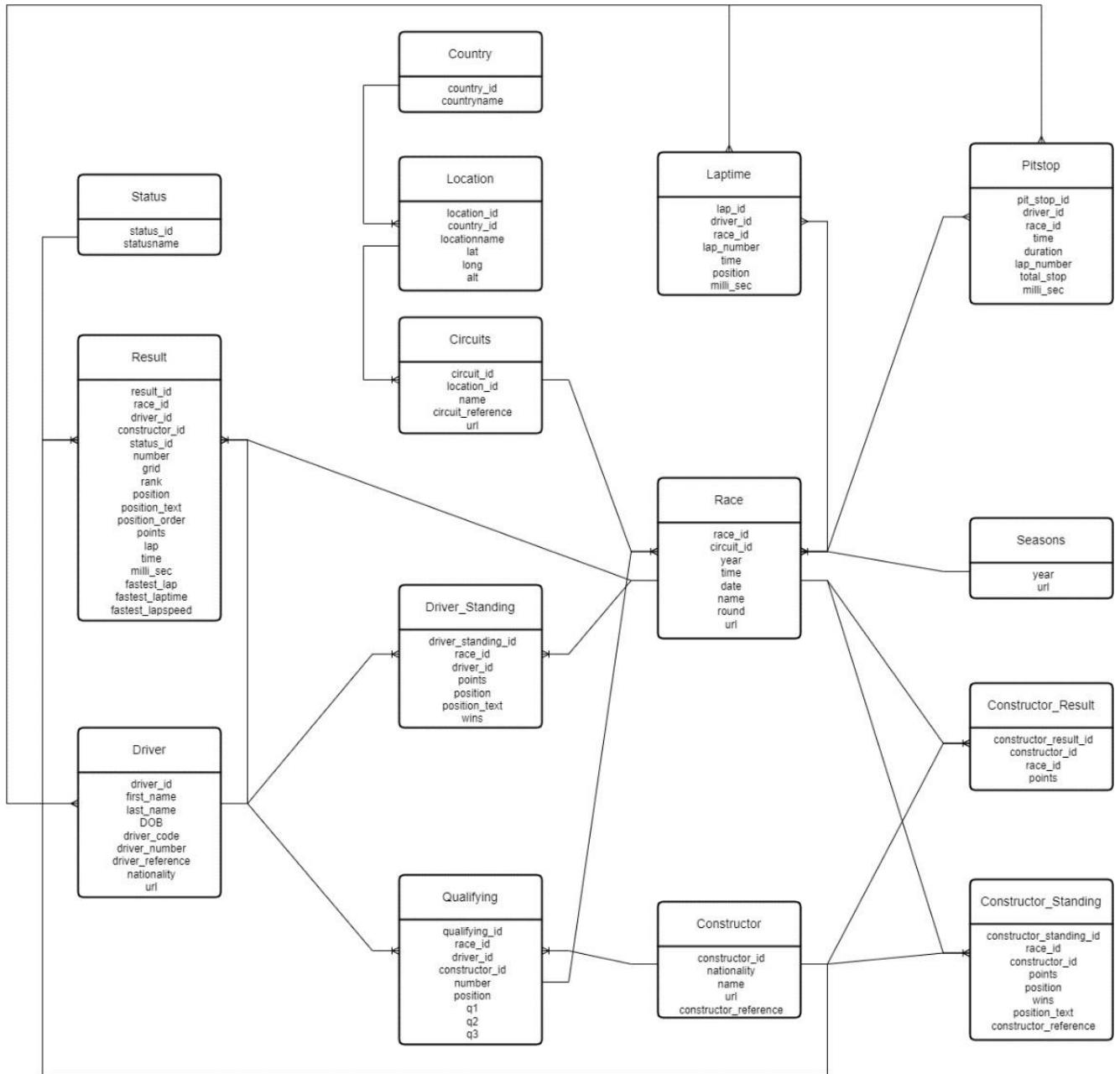
ERD:



Relational Model :



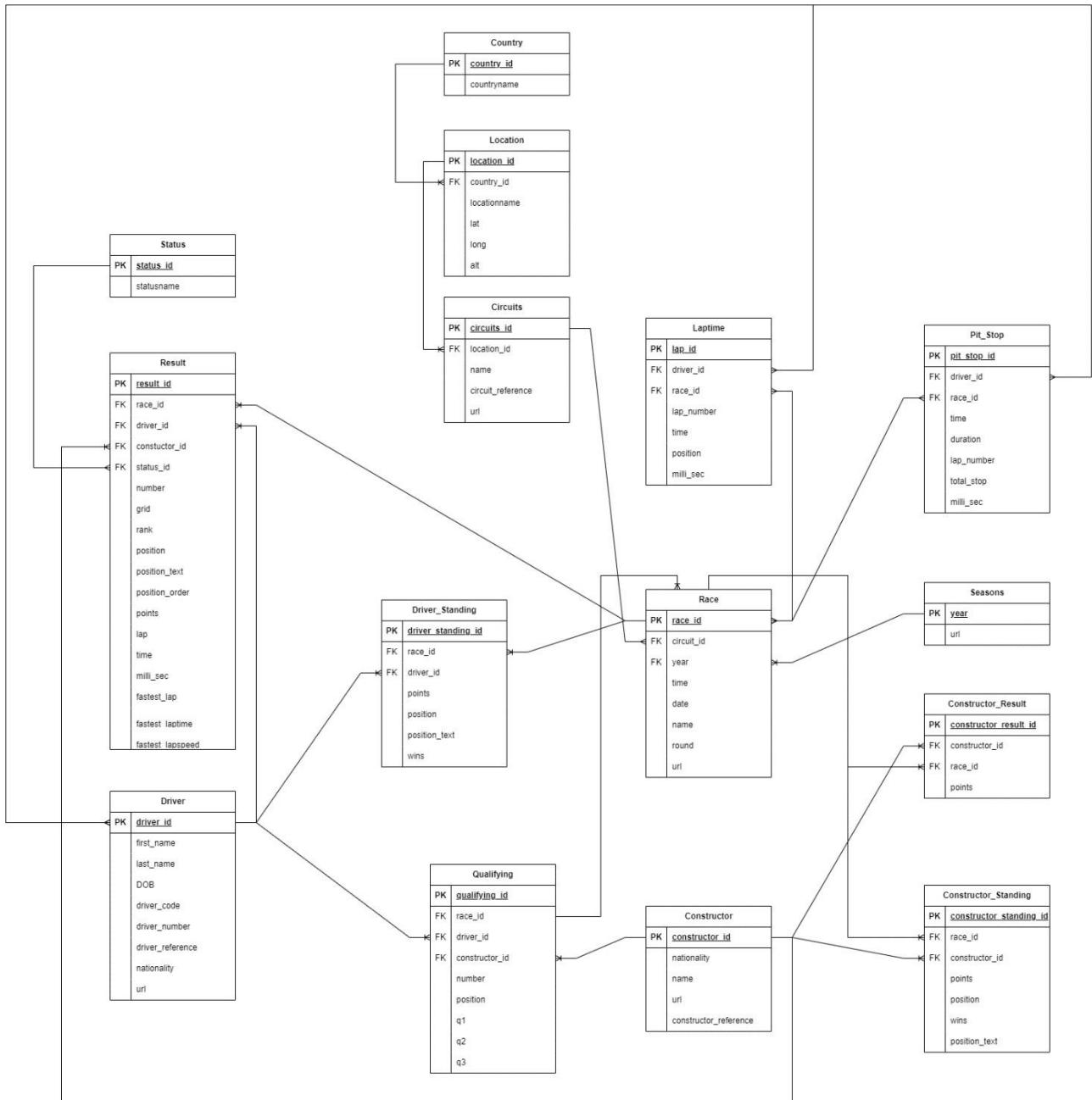
Data Warehouse Model:



Conceptual Model

Possible Hierarchies:

1. Country → Location → Circuits
2. Driver → Driver Standing
3. Constructor → Constructor Standing



Logical Model

OLAP OPERATIONS FOR ANALYSIS :

1. Select avg number of total stops taken by driver whose first name and last name is Nico Rosberg in year 2008.

OUTPUT 1 \leftarrow DICE (F1, driver.first_name = ‘Nico AND driver.last_name = ‘Rosberg’ AND race.year = 2008)

OUTPUT \leftarrow ROLLUP * (OUTPUT 1, F1 \rightarrow F1, AVG (total_stop) AS avg_stops)

2. Total number of Constructor who took part in year 2009 at Germany.

OUTPUT 2 \leftarrow DICE (F1, race.year = 2009 AND circuit.country = ‘Germany’)

OUTPUT \leftarrow ROLLUP * (OUTPUT 2, F1 \rightarrow F1, COUNT (constructor.id) AS total_constructor)

3. SELECT drivers who qualified in year 2009 at Italian Grand Prix.

OUTPUT \leftarrow DICE (F1, race.year = 2009 AND circuit.name = “Italian Grand Prix”)

4. SELECT and sort the ranks of drivers in Ascending order for the race played at Turkish Grand Prix in year 2008.

OUTPUT 4 \leftarrow DICE (F1, race.year = 2008 AND race.name = ‘Turkish Grand Prix’)

OUTPUT \leftarrow SORT (OUTPUT 4, Result, Rank [ASC])

5. Provide the minimum fastest lap speed for the races played in Malaysia.

OUTPUT 5 \leftarrow DICE (F1, circuit.country = ‘Malaysia’)

OUTPUT \leftarrow ROLLUP* (OUTPUT 5, F1 \rightarrow F1, MIN (fastest_lap) as fastest_lap)

6. Find out total race having race name Bahrain Grand Prix

OUTPUT 6 \leftarrow SLICE (F1, Race, race.name = ‘Bahrain Grand Prix’)

OUTPUT \leftarrow ROLLUP* (OUTPUT 6, Race \rightarrow Race, COUNT (race_id))

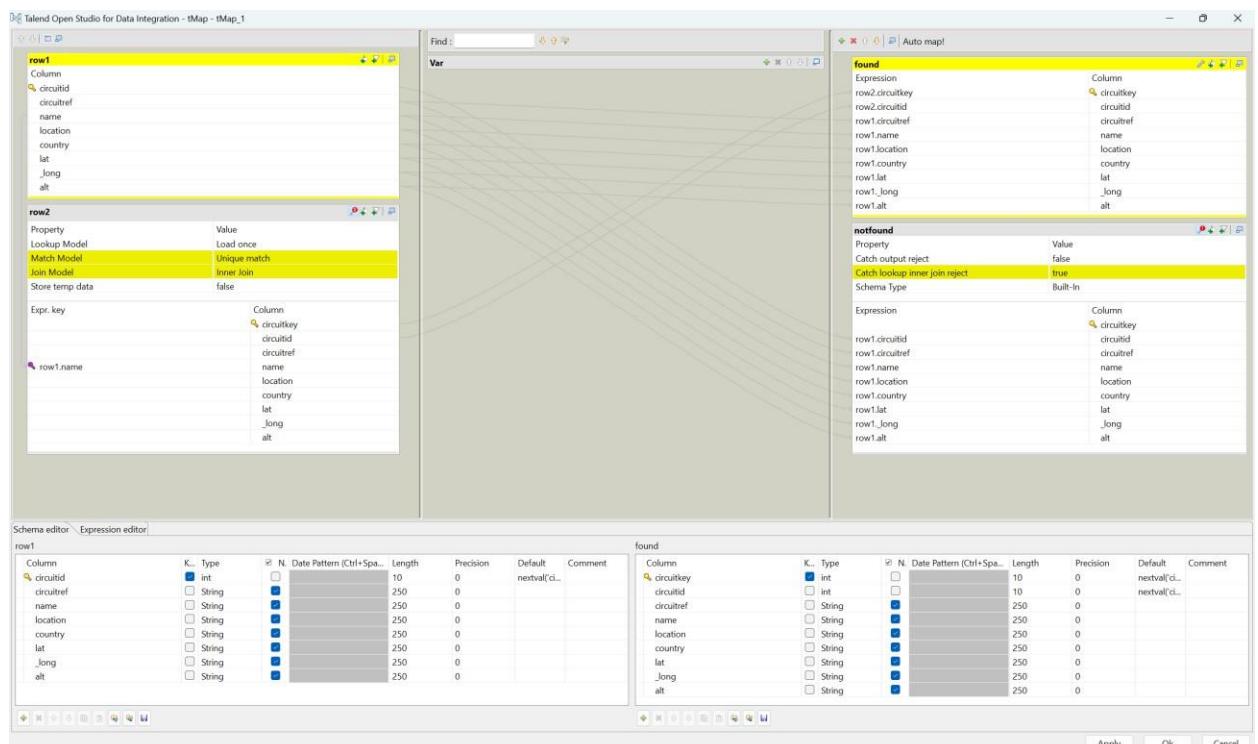
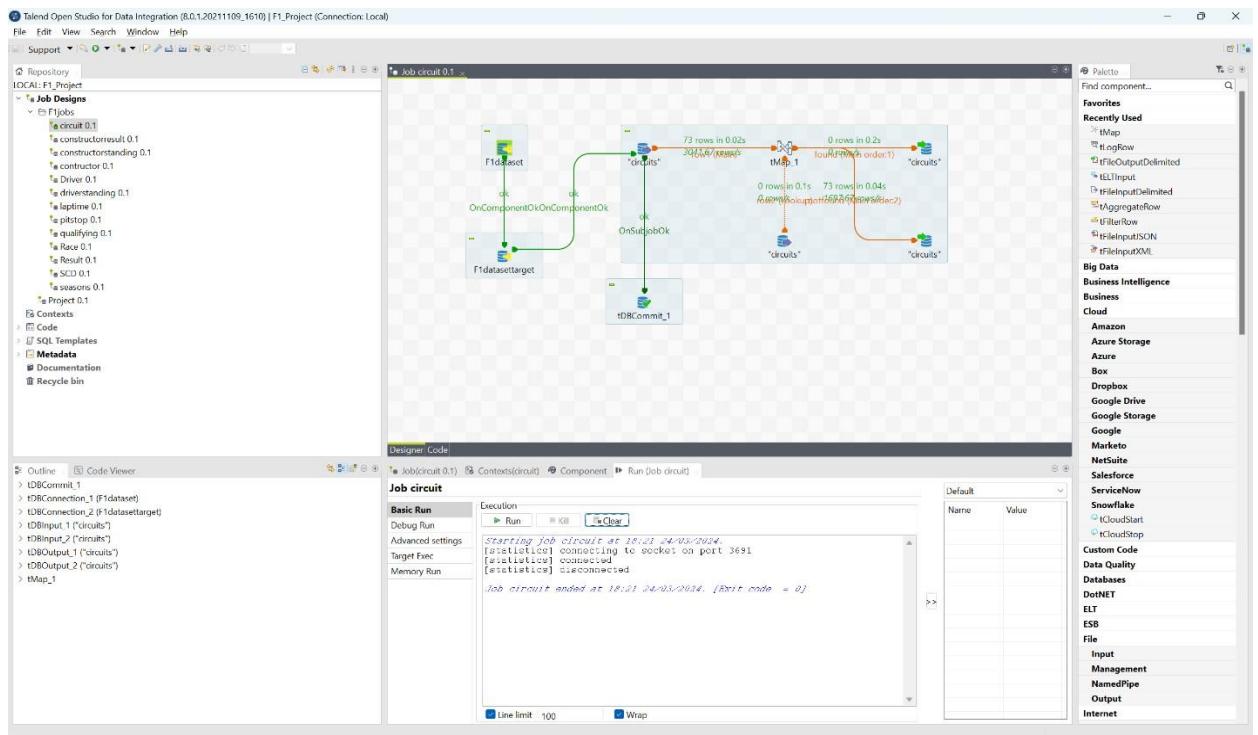
7. Select the first qualifying driver in race Australian Grand Prix

OUTPUT 7 ← DICE (F1, race.name = ‘Australian Grand Prix’)

OUTPUT ← ROLLUP* (OUTPUT 7, Qualifying → Qualifying, MIN(q1) ASfirst_qualifier)

Talend Implementation control and data flows

1. Circuits Dimension



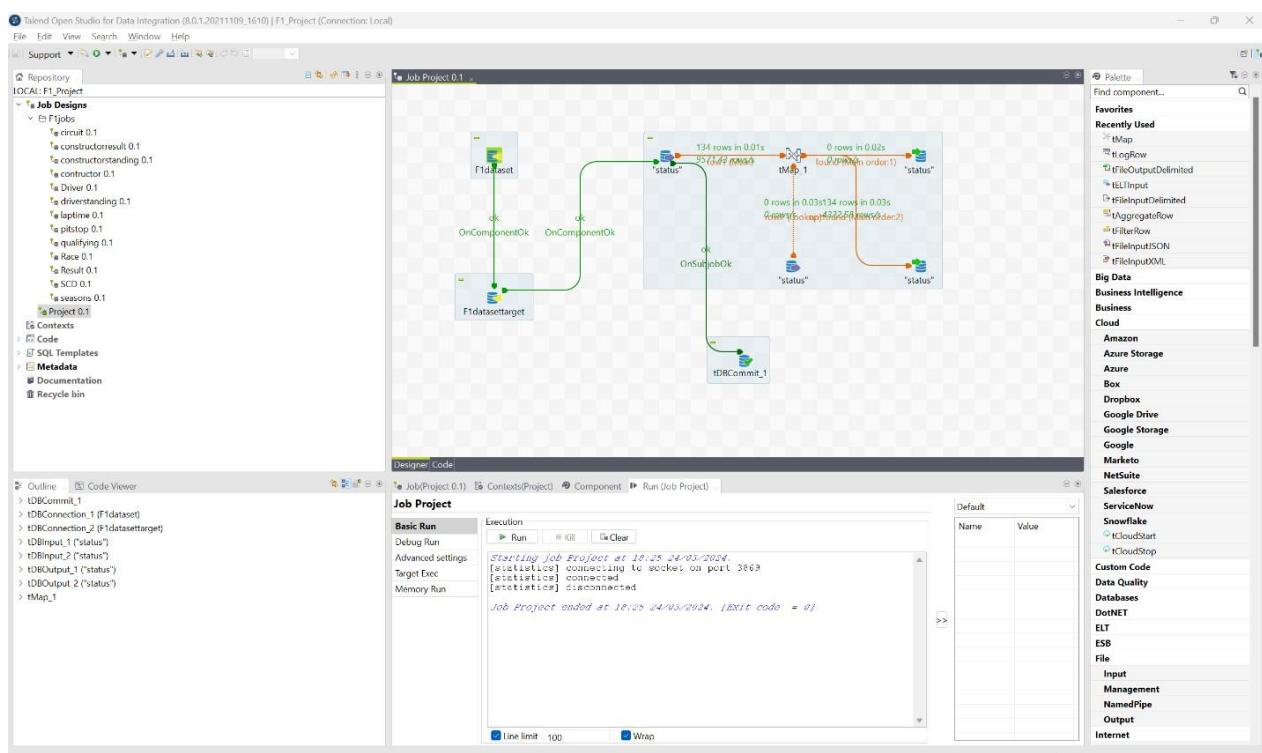
Circuits Dimension Output

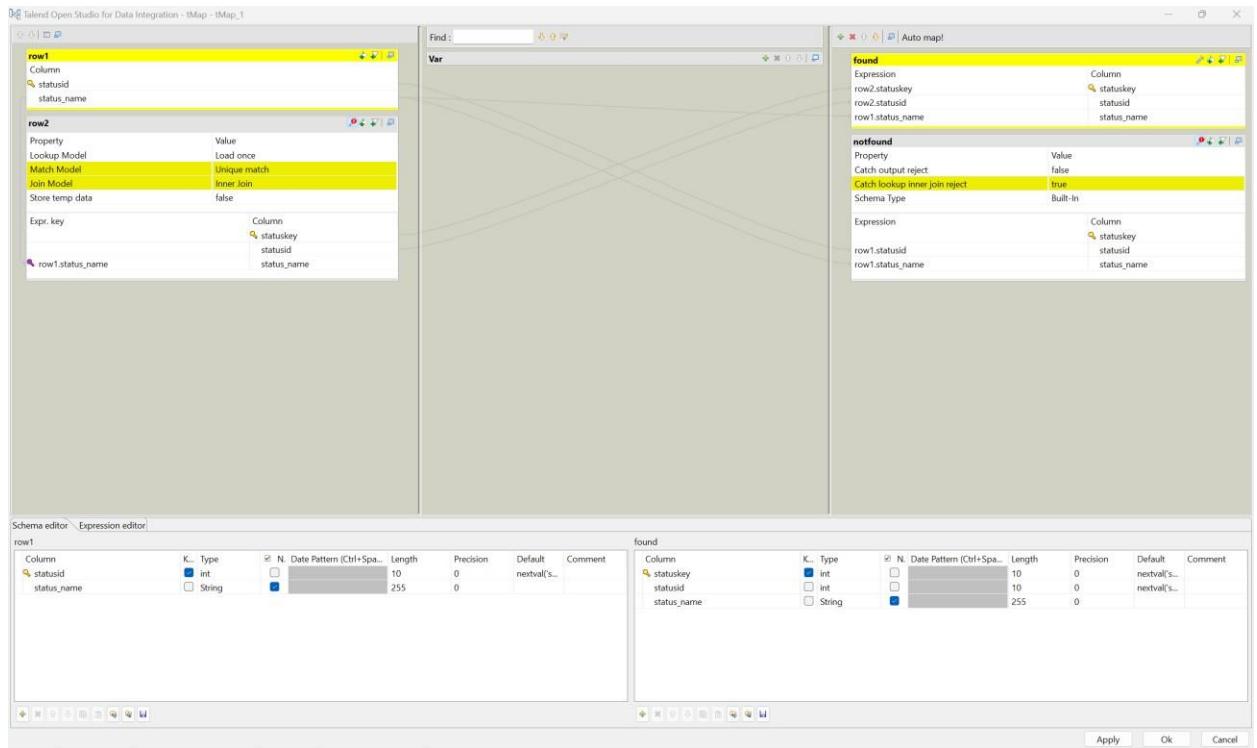
Data Output Messages Notifications

	circuitkey [PK] integer	circuitid integer	circuitref character varying (250)	name character varying (250)	location character varying (250)	country character varying (250)	lat character varying (250)	long character
1	1	1	albert_park	Albert Park Grand Prix Circuit	Melbourne	Australia	-37.8497	144.9
2	2	2	sepang	Sepang International Circuit	Kuala Lumpur	Malaysia	2.76083	101.7
3	3	3	bahrain	Bahrain International Circuit	Sakhir	Bahrain	26.0325	50.51
4	4	4	catalunya	Circuit de Barcelona-Catalunya	Montmel	Spain	41.57	2.261
5	5	5	istanbul	Istanbul Park	Istanbul	Turkey	40.9517	29.40
6	6	6	monaco	Circuit de Monaco	MonteCarlo	Monaco	43.7347	7.420
7	7	7	villeneuve	Circuit Gilles Villeneuve	Montreal	Canada	45.5	-73.5
8	8	8	magny_cours	Circuit de Nevers Magny-Cours	Magny Cours	France	46.8642	3.163
9	9	9	silverstone	Silverstone Circuit	Silverstone	UK	52.0786	-1.01
10	10	10	hockenheimring	Hockenheimring	Hockenheim	Germany	49.3278	8.565
11	11	11	hungaroring	Hungaroring	Budapest	Hungary	47.5789	19.24
12	12	12	valencia	Valencia Street Circuit	Valencia	Spain	39.4589	-0.33

Total rows: 73 of 73 Query complete 00:00:00.122 Ln 1, Col 23

2. Status Dimension





Status Dimension Output

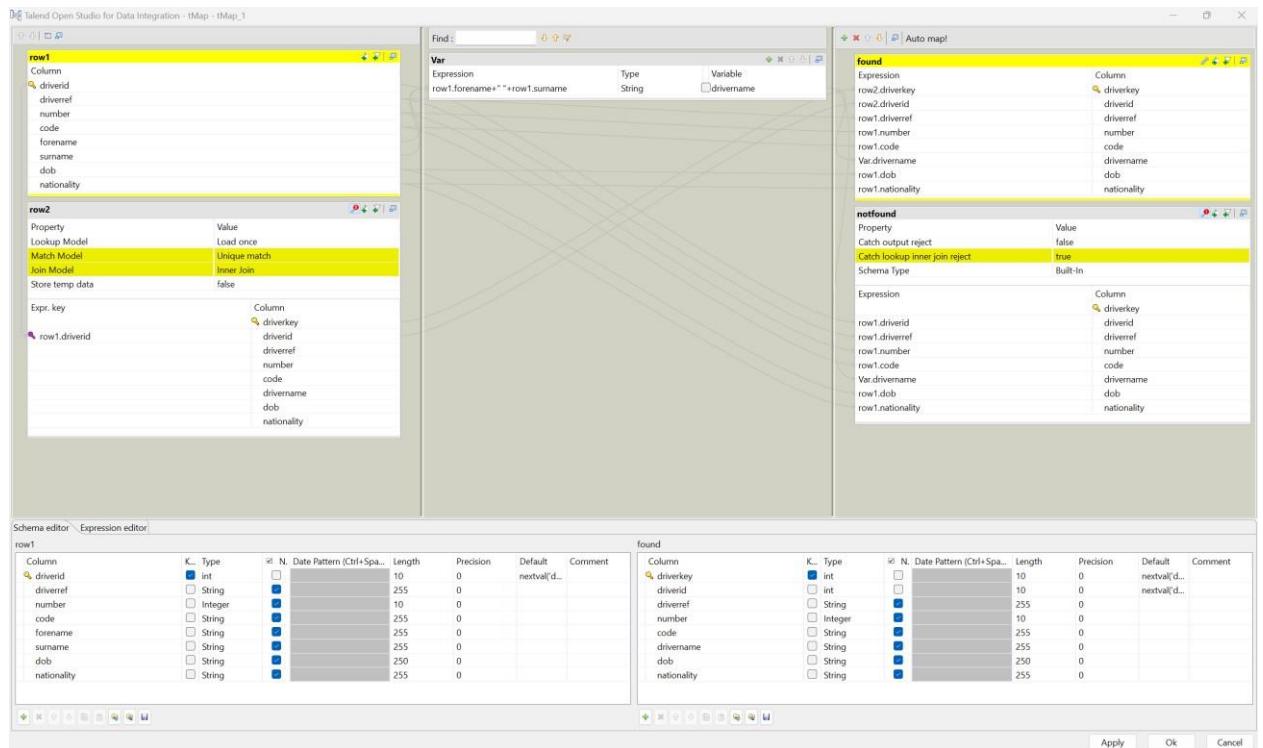
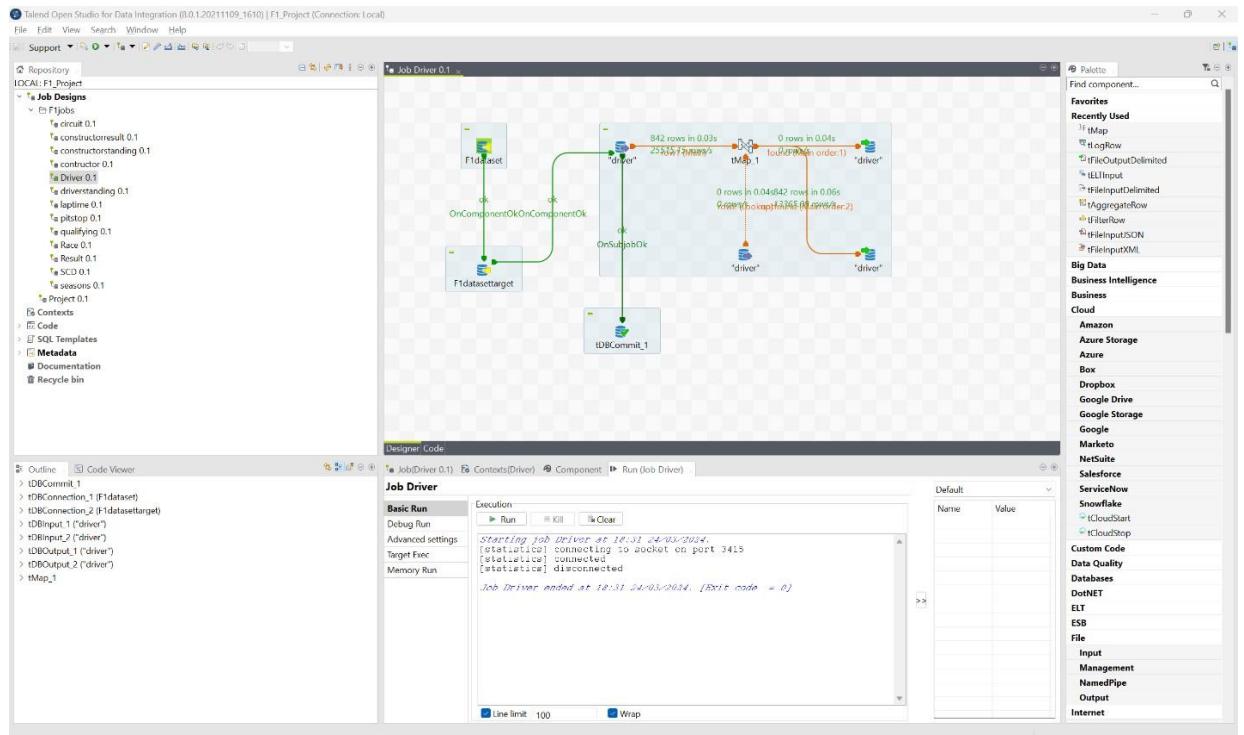
Data Output Messages Notifications

File Database Flat File XML CSV Excel PDF Word PowerPoint

Total rows: 134 of 134 Query complete 00:00:00.149 Ln 1, Col 21

	statuskey [PK] integer	statusid integer	status_name character varying (255)
1	1	1	Finished
2	2	2	Disqualified
3	3	3	Accident
4	4	4	Collision
5	5	5	Engine
6	6	6	Gearbox
7	7	7	Transmission
8	8	8	Clutch
9	9	9	Hydraulics
10	10	10	Electrical
11	11	11	+1 Lap
12	12	12	+2 Laps

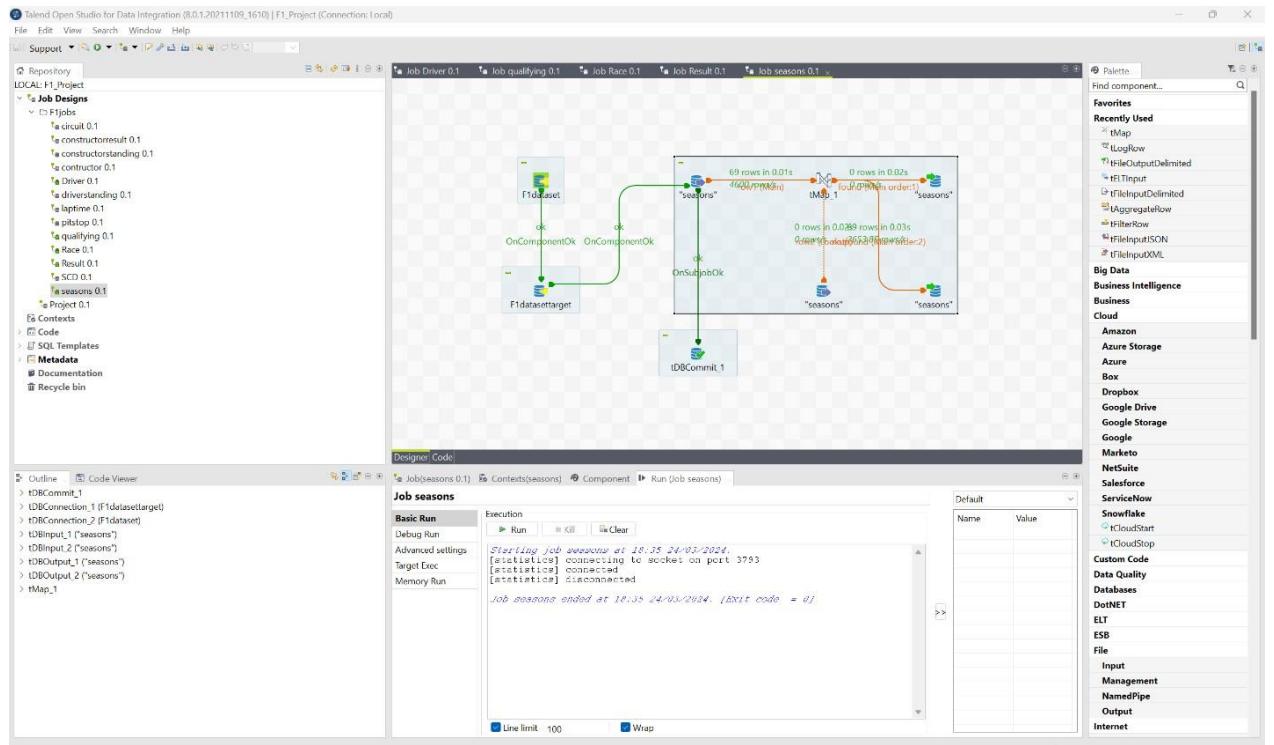
3. Driver Dimension

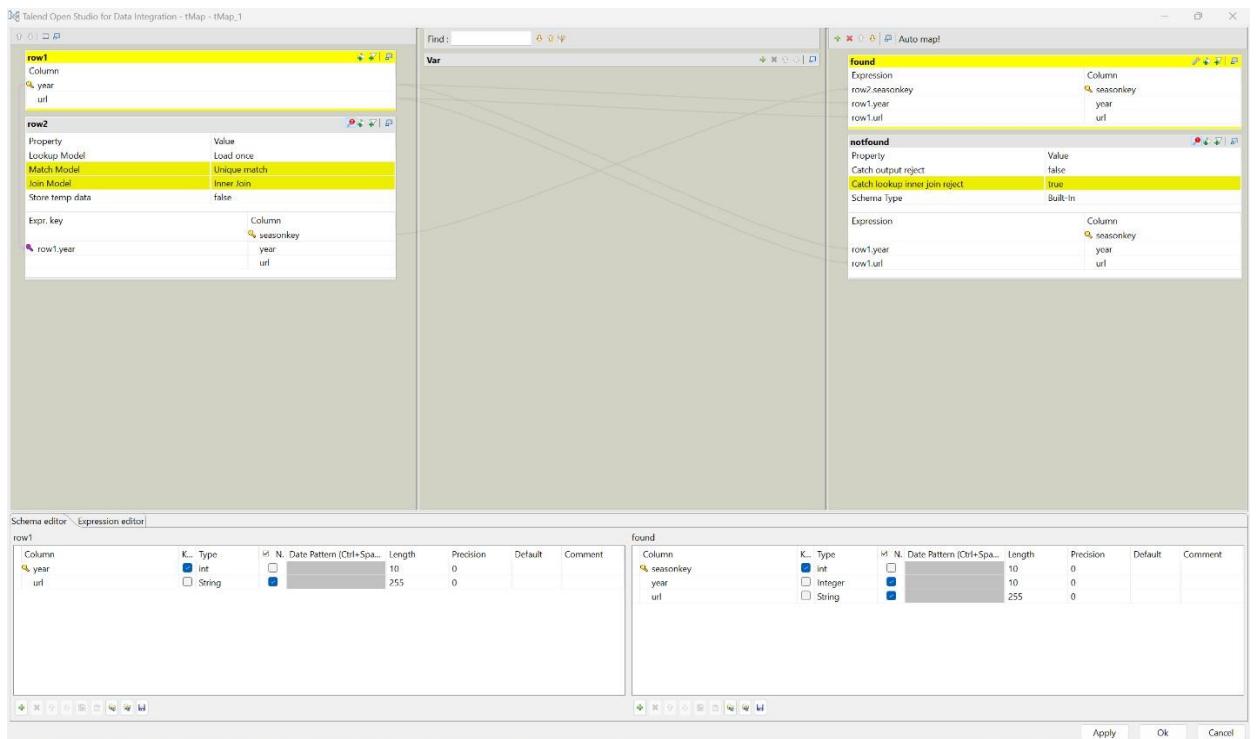


Driver Dimension Output

Data Output		Messages		Notifications				X
	driverkey [PK] integer	driverid integer	driverref character varying (255)	number integer	code character varying (255)	drivername character varying (255)	dob character varying (250)	nationality character varying (255)
1		1	hamilton		44 HAM	Lewis Hamilton	7/1/1985	British
2		2	heidfeld		[null] HEI	Nick Heidfeld	10/5/1977	German
3		3	rosberg		6 ROS	Nico Rosberg	27/06/1985	German
4		4	alonso		14 ALO	Fernando Alonso	29/07/1981	Spanish
5		5	kovalainen		[null] KOV	Heikki Kovalainen	19/10/1981	Finnish
6		6	nakajima		[null] NAK	Kazuki Nakajima	11/1/1985	Japanese
7		7	bourdais		[null] BOU	Sbastien Bourdais	28/02/1979	French
8		8	raikkonen		7 RAI	Kimi Rnen	17/10/1979	Finnish
9		9	kubica		[null] KUB	Robert Kubica	7/12/1984	Polish
10		10	glock		[null] GLO	Timo Glock	18/03/1982	German
11		11	sato		[null] SAT	Takuma Sato	28/01/1977	Japanese
12		12	piquet_ir		[null] PIQ	Nelson Piquet Jr	25/07/1985	Brazilian

4. Seasons Dimension





Season Dimension Output

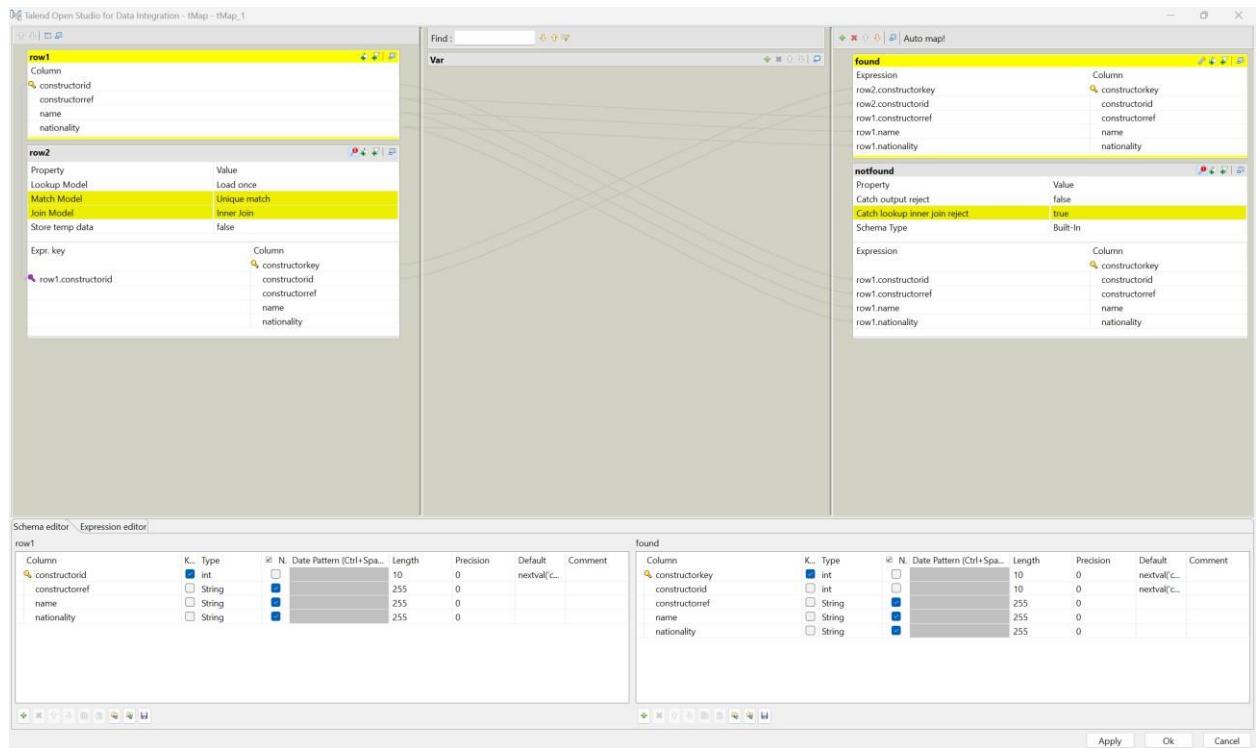
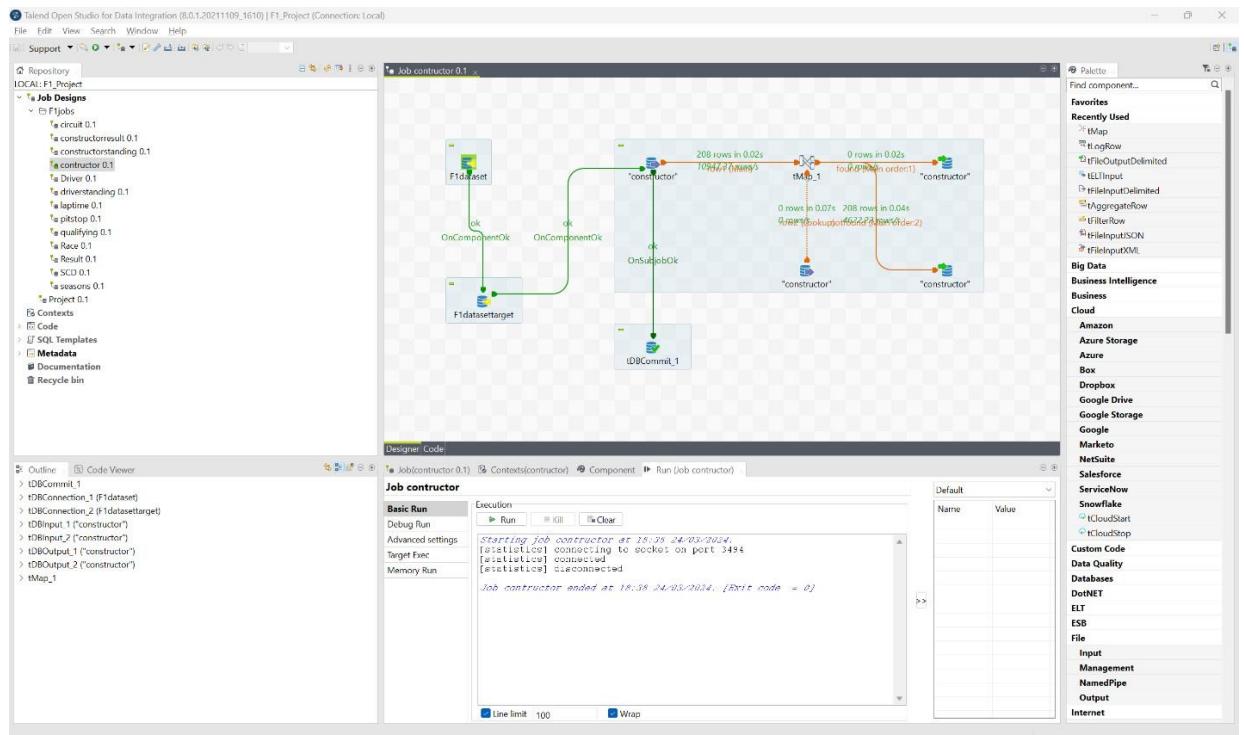
Data Output Messages Notifications

seasonkey [PK] integer year integer url character varying (255)

	seasonkey	year	url
1	1	2009	http://en.wikipedia.org/wiki/2009_Formula_One_season
2	2	2008	http://en.wikipedia.org/wiki/2008_Formula_One_season
3	3	2007	http://en.wikipedia.org/wiki/2007_Formula_One_season
4	4	2006	http://en.wikipedia.org/wiki/2006_Formula_One_season
5	5	2005	http://en.wikipedia.org/wiki/2005_Formula_One_season
6	6	2004	http://en.wikipedia.org/wiki/2004_Formula_One_season
7	7	2003	http://en.wikipedia.org/wiki/2003_Formula_One_season
8	8	2002	http://en.wikipedia.org/wiki/2002_Formula_One_season
9	9	2001	http://en.wikipedia.org/wiki/2001_Formula_One_season
10	10	2000	http://en.wikipedia.org/wiki/2000_Formula_One_season
11	11	1999	http://en.wikipedia.org/wiki/1999_Formula_One_season
12	12	1998	http://en.wikipedia.org/wiki/1998_Formula_One_season

Total rows: 69 of 69 Query complete 00:00:00.119 Ln 1, Col 22

5. Constructor Dimension



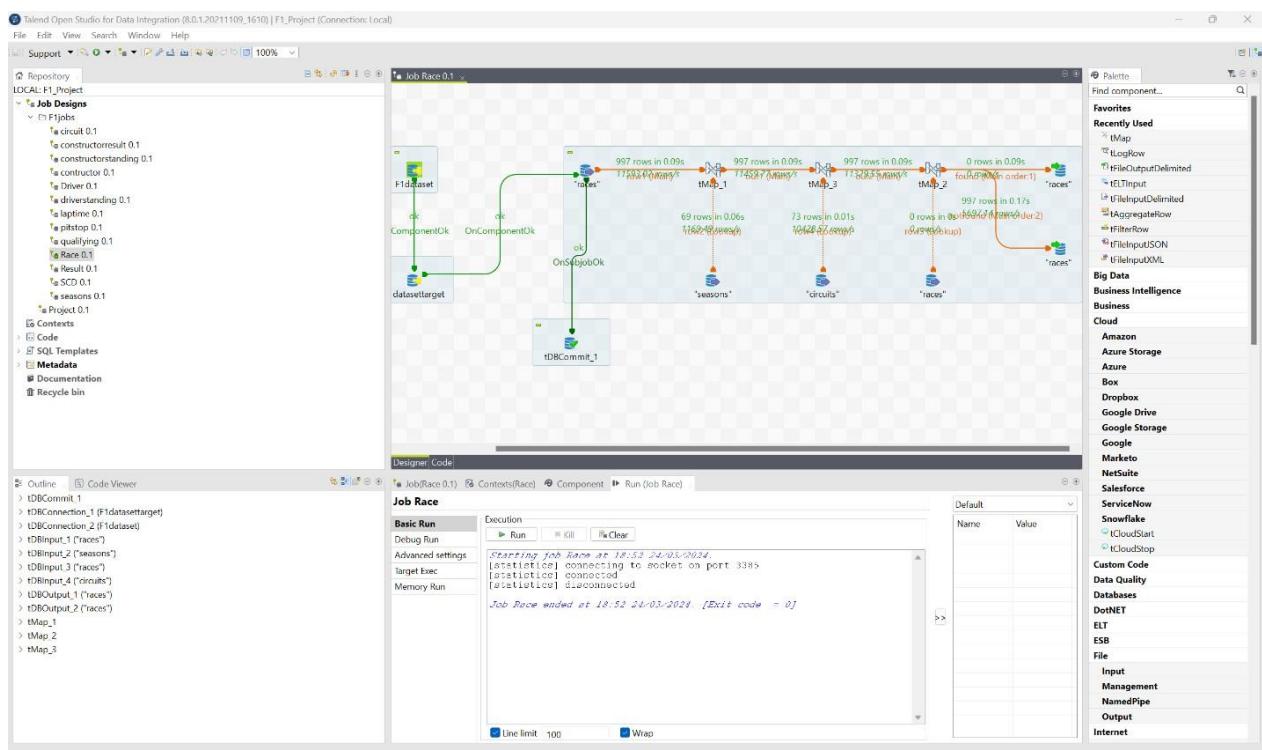
Constructor Dimension Output

Data Output Messages Notifications

	constructorkey [PK] integer	constructorid integer	constructorref character varying (255)	name character varying (255)	nationality character varying (255)
1	1	1	mclaren	McLaren	British
2	2	2	bmw_saucer	BMW Sauber	German
3	3	3	williams	Williams	British
4	4	4	renault	Renault	French
5	5	5	toro_rosso	Toro Rosso	Italian
6	6	6	ferrari	Ferrari	Italian
7	7	7	toyota	Toyota	Japanese
8	8	8	super_aguri	Super Aguri	Japanese
9	9	9	red_bull	Red Bull	Austrian
10	10	10	force_india	Force India	Indian
11	11	11	honda	Honda	Japanese
12	12	12	spyker	Spyker	Dutch

Total rows: 208 of 208 Query complete 00:00:00.120 Ln 1, Col 26

6. Races Dimension



Talend Open Studio for Data Integration - tMap - tMap_2

out2

Column
circuitkey
round
name
date
time
raceid
seasonkey

row3

Property Value
Lookup Model Load once
Match Model Unique match
Join Model Inner Join
Store temp data false

Expr. key Column
out2.raceid racekey
raceid seasonkey
round circuitkey
name name
date date
time time

found

Column
racekey
raceid
seasonkey
round
circuitkey
name
date
time

notfound

Property Value
Catch output reject false
Catch lookup inner join reject true
Schema Type Built-In

Expression Column
out2.raceid racekey
out2.seasonkey seasonkey
out2.round round
out2.circuitkey circuitkey
out2.name name
out2.date date
out2.time time

Schema editor \ Expression editor

out2

Column	K... Type	N. Date Pattern (Ctrl+Space)	Length	Precision	Default	Comment
circuitkey	Integer	dd-MM-yyyy	10	0		
round	String		10	0		
name	String		250	0		
date	Date		10	0		
time	String		255	0		
raceid	Integer	dd-MM-yyyy	13	0		
seasonkey	Integer		250	0		

found

Column	K... Type	N. Date Pattern (Ctrl+Space)	Length	Precision	Default	Comment
racekey	int	dd-MM-yyyy	10	0	nextval('r...',	
raceid	Integer		10	0		
seasonkey	String		250	0		
round	String		10	0		
circuitkey	int		255	0		
name	String		255	0		
date	Date		13	0		
time	String		250	0		

Apply Ok Cancel

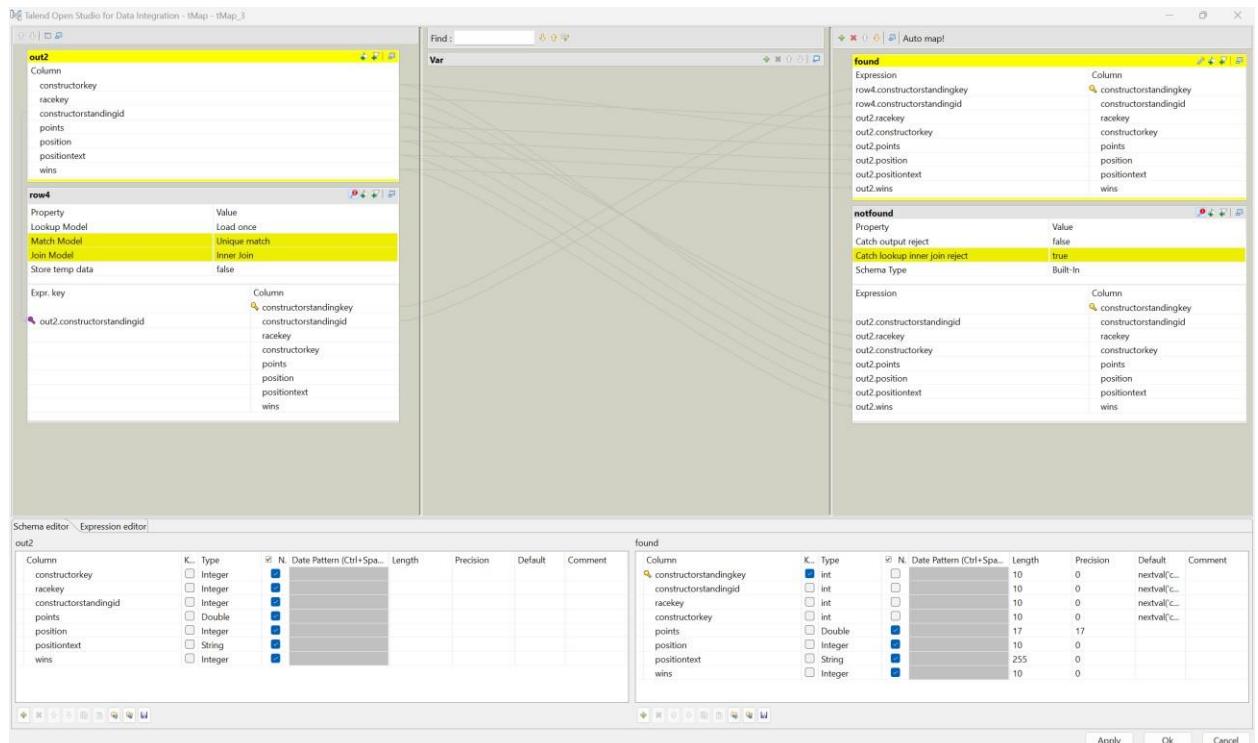
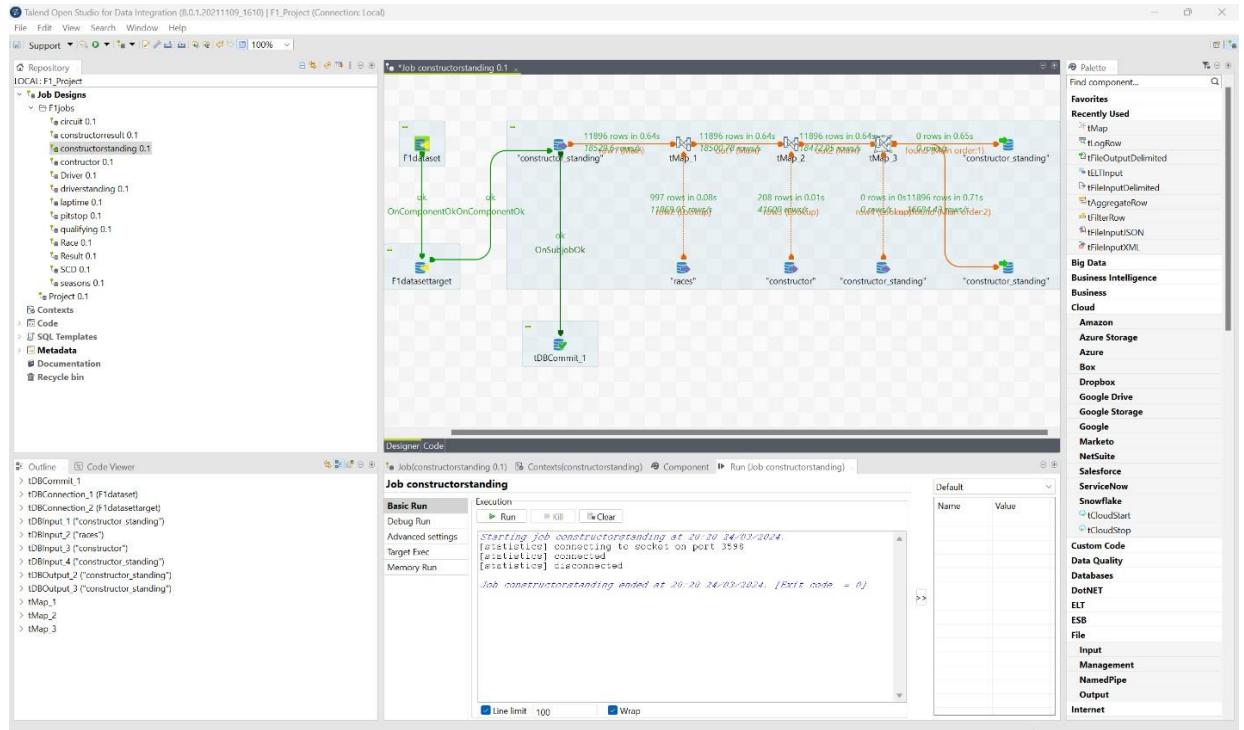
Races Dimension Output

Data Output Messages Notifications

	racekey [PK] integer	raceid integer	seasonkey integer	round character varying (250)	circuitkey integer	name character varying (255)	date date	time character varying (250)
1	1	1	1	1		Australian Grand Prix	2009-03-29	6:00:00
2	2	2		1	2	Malaysian Grand Prix	2009-04-05	9:00:00
3	3	3		1	3	Chinese Grand Prix	2009-04-19	7:00:00
4	4	4		1	4	Bahrain Grand Prix	2009-04-26	12:00:00
5	5	5		1	5	Spanish Grand Prix	2009-05-10	12:00:00
6	6	6		1	6	Monaco Grand Prix	2009-05-24	12:00:00
7	7	7		1	7	Turkish Grand Prix	2009-06-07	12:00:00
8	8	8		1	8	British Grand Prix	2009-06-21	12:00:00
9	9	9		1	9	German Grand Prix	2009-07-12	12:00:00
10	10	10		1	10	Hungarian Grand Prix	2009-07-26	12:00:00
11	11	11		1	11	European Grand Prix	2009-08-23	12:00:00
12	12	12		1	12	Belgian Grand Prix	2009-08-30	12:00:00

Total rows: 997 of 997 Query complete 00:00:00.216 Ln 1, Col 20

7. Constructor Standing Dimension



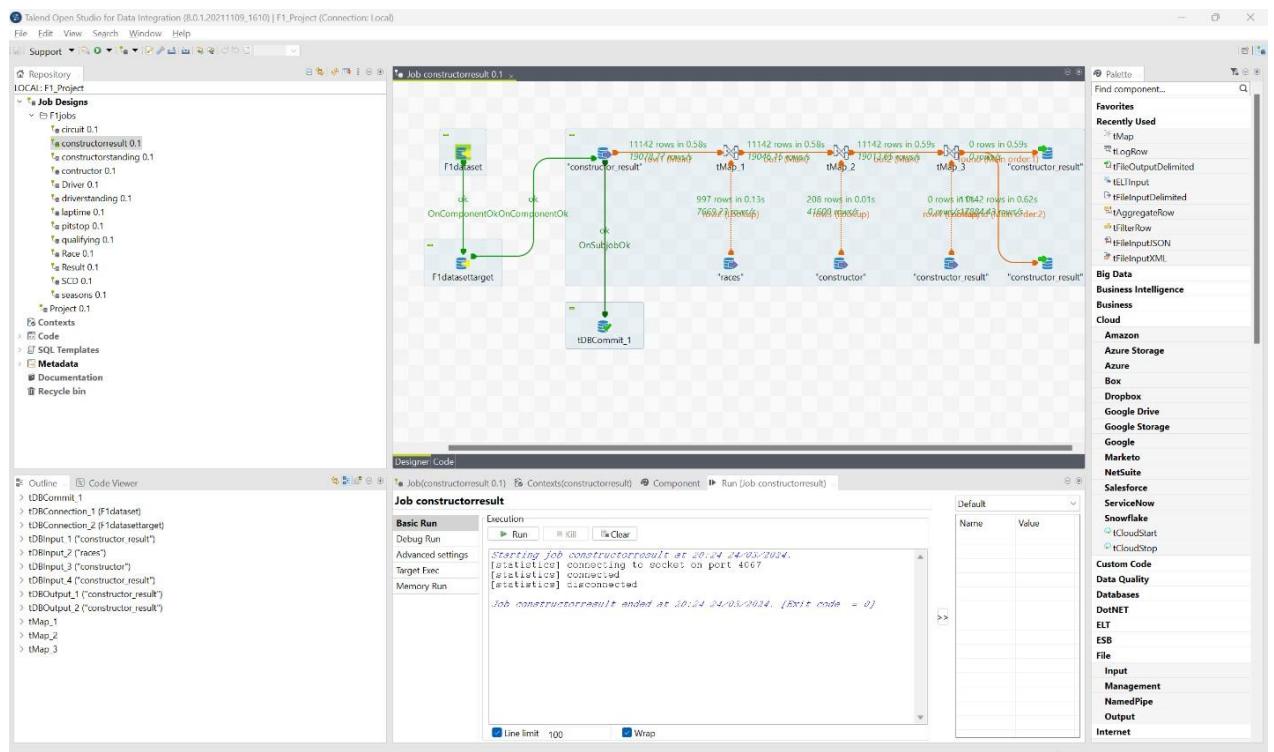
Constructor Standing Dimension Output

Data Output Messages Notifications

	constructorstandingkey [PK] integer	constructorstandingid integer	racekey integer	constructorkey integer	points double precision	position integer	positiontext character varying (255)	wins integer
1		1	18	1	14	1	1	1
2		2	18	2	8	3	3	0
3		3	18	3	9	2	2	0
4		4	18	4	5	4	4	0
5		5	18	5	2	5	5	0
6		6	18	6	1	6	6	0
7		7	19	1	24	1	1	1
8		8	19	2	19	2	2	0
9		9	19	3	9	4	4	0
10		10	19	4	6	5	5	0
11		11	19	5	2	8	8	0
12		12	19	6	11	3	3	1

Total rows: 1000 of 11896 Query complete 00:00:00.155 Ln 1, Col 35

8. Constructor Result Dimension



Talend Open Studio for Data Integration - tMap - tMap_3

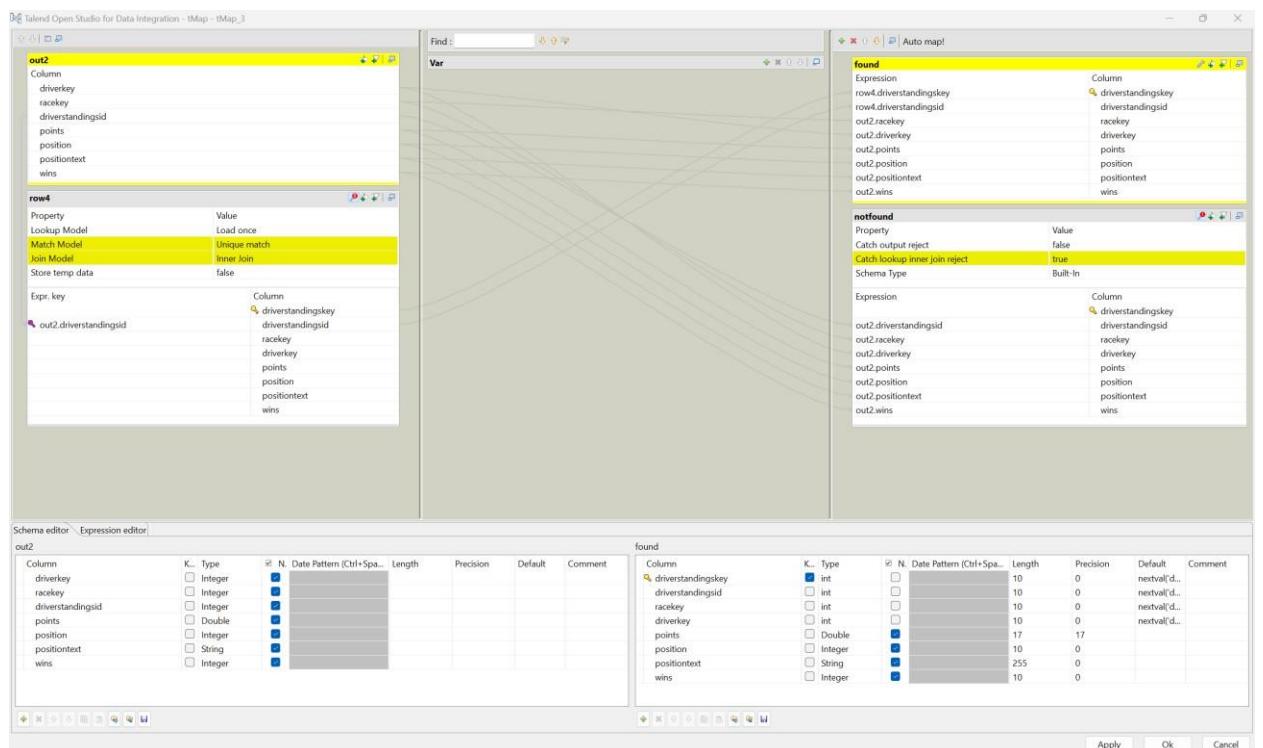
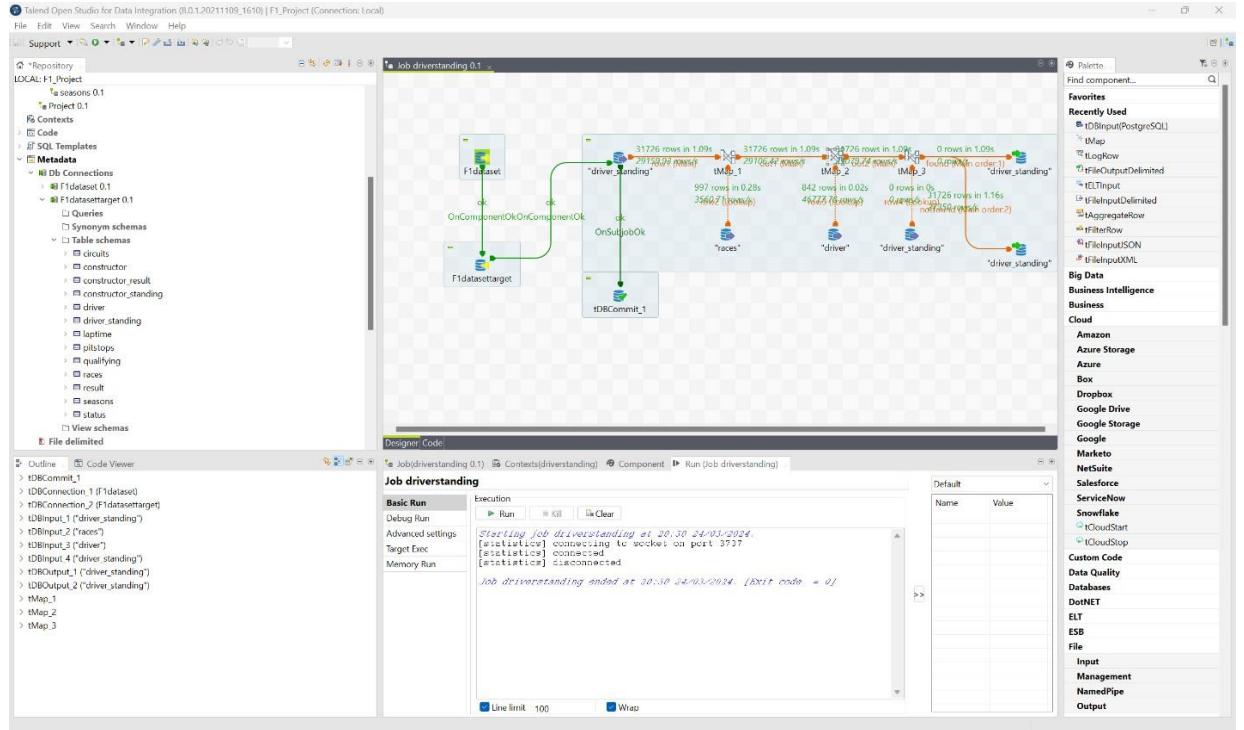
Constructor Result Dimension Output

Data Output Messages Notifications

	constructorresultkey [PK] integer	constructorresultid integer	racekey integer	constructorkey integer	points double precision	
1		1	18	1	14	
2		2	18	2	8	
3		3	18	3	9	
4		4	18	4	5	
5		5	18	5	2	
6		6	18	6	1	
7		7	18	7	0	
8		8	18	8	0	
9		9	18	9	0	
10		10	18	10	0	
11		11	18	11	0	
12		12	19	6	10	

Total rows: 1000 of 11142 Query complete 00:00:00.142 Ln 1, Col 33

9. Driver Standing Dimension



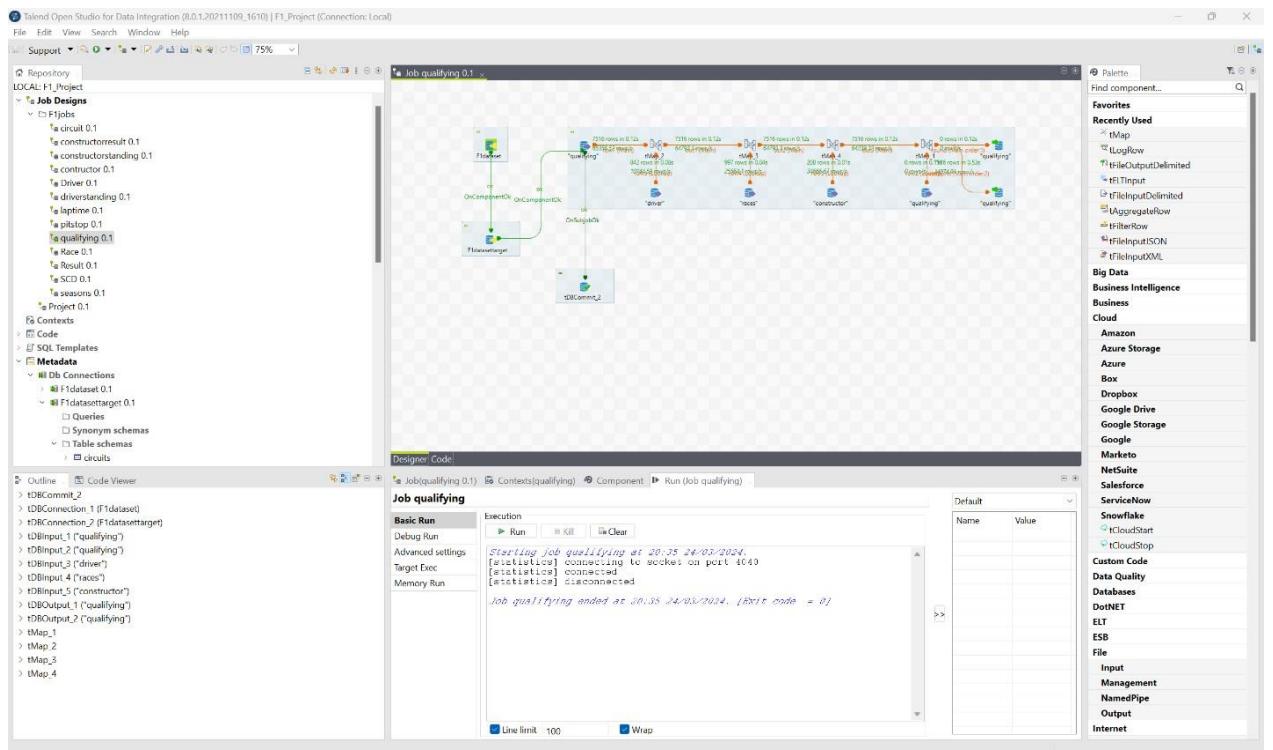
Driver Standing Dimension Output

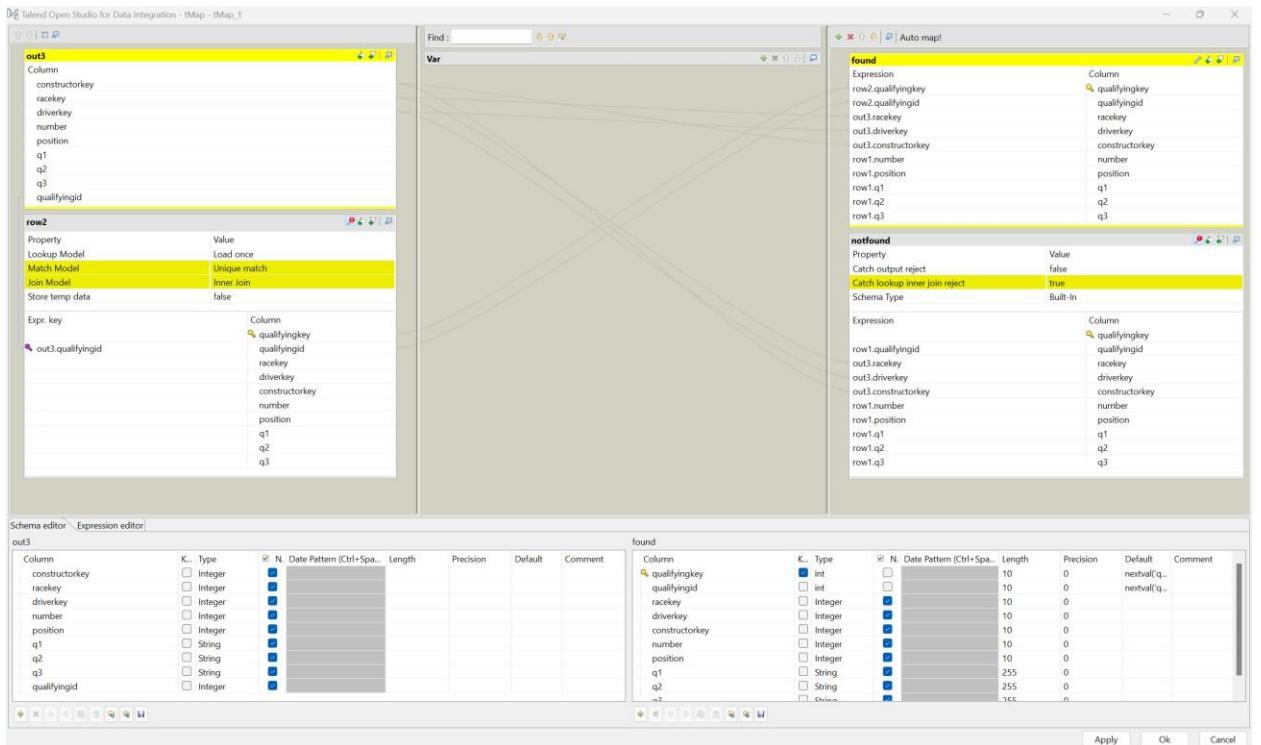
Data Output Messages Notifications

	driverstandingskey [PK] integer	driverstandingsid integer	racekey integer	driverkey integer	points double precision	position integer	positiontext character varying (255)	wins integer
1	1	1	18	1	10	1	1	1
2	2	2	18	2	8	2	2	0
3	3	3	18	3	6	3	3	0
4	4	4	18	4	5	4	4	0
5	5	5	18	5	4	5	5	0
6	6	6	18	6	3	6	6	0
7	7	7	18	7	2	7	7	0
8	8	8	18	8	1	8	8	0
9	9	9	19	1	14	1	1	1
10	10	10	19	2	11	3	3	0
11	11	11	19	3	6	6	6	0
12	12	12	19	4	6	7	7	0

Total rows: 1000 of 31726 Query complete 00:00:00.199 Ln 1, Col 30

10. Qualifying Dimension





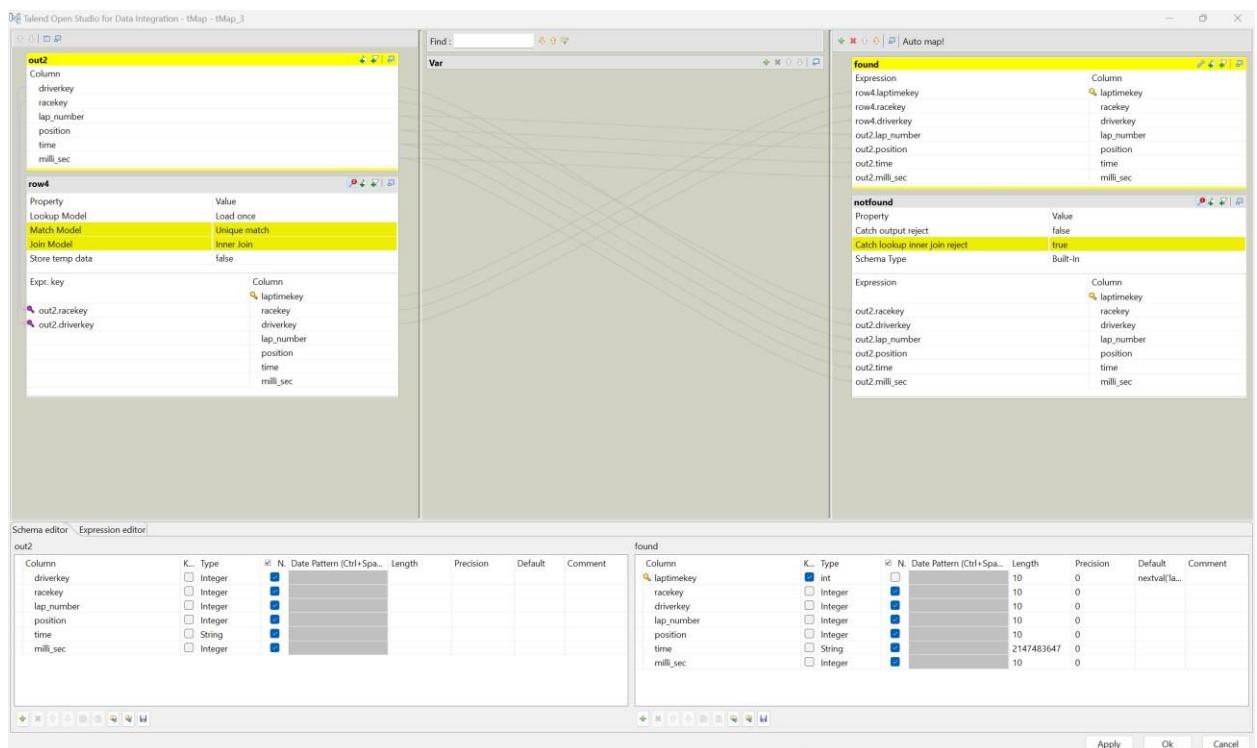
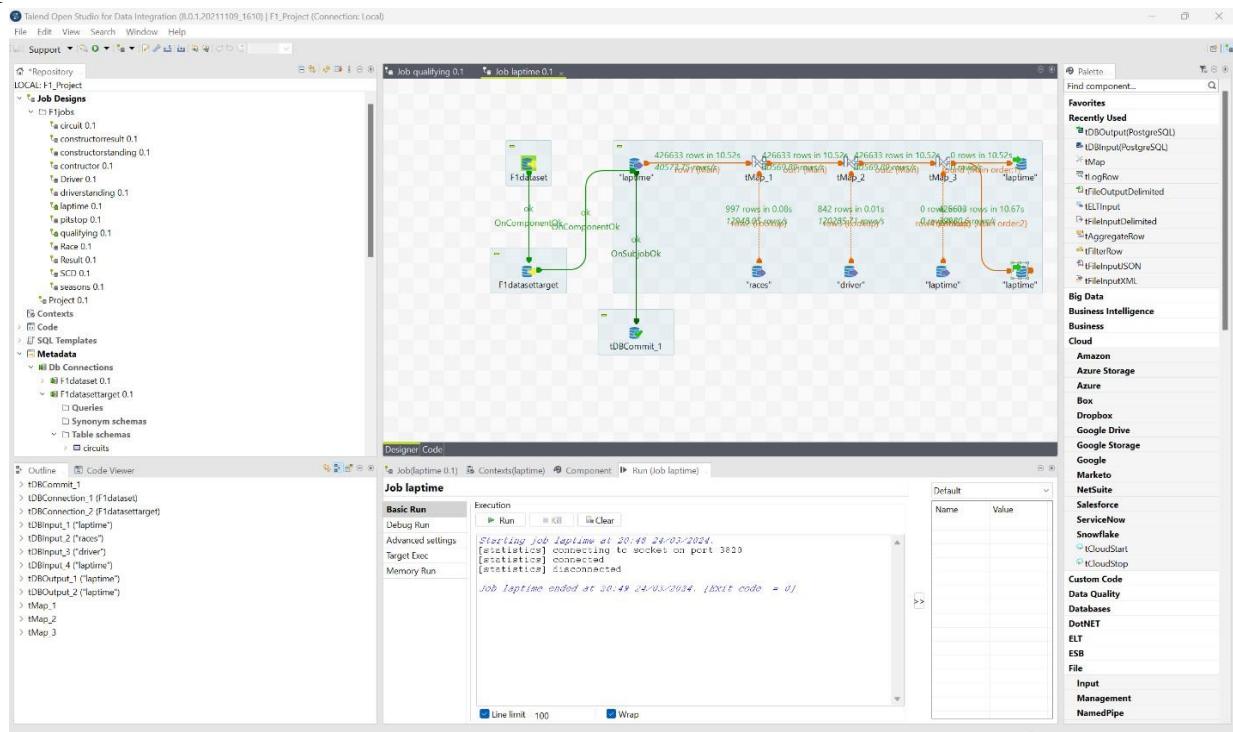
Qualifying Dimension Output

Data Output Messages Notifications

	qualifyingkey [PK] integer	qualifyingid integer	racekey integer	driverkey integer	constructorkey integer	number integer	position integer	q1 character varying (255)	q2 character varying (255)	q3 character varying (255)
1	1	1	18	1	1	22	1	01:26.6	01:25.2	01:26.7
2	2	2	18	9	2	4	2	01:26.1	01:25.3	01:26.9
3	3	3	18	5	1	23	3	01:25.7	01:25.5	01:27.1
4	4	4	18	13	6	2	4	01:26.0	01:25.7	01:27.2
5	5	5	18	2	2	3	5	01:26.0	01:25.5	01:27.2
6	6	6	18	15	7	11	6	01:26.4	01:26.1	01:28.5
7	7	7	18	3	3	7	7	01:26.3	01:26.1	01:28.7
8	8	8	18	14	9	9	8	01:26.4	01:26.1	01:29.0
9	9	9	18	10	7	12	9	01:26.9	01:26.2	01:29.6
10	10	10	18	20	5	15	10	01:26.7	01:25.8	[null]
11	11	11	18	22	11	17	11	01:26.4	01:26.2	[null]
12	12	12	18	4	4	5	12	01:26.9	01:26.2	[null]

Total rows: 1000 of 7516 Query complete 00:00:00.180 Ln 1, Col 25

11. Lap time Dimension



Lap time Dimension Output

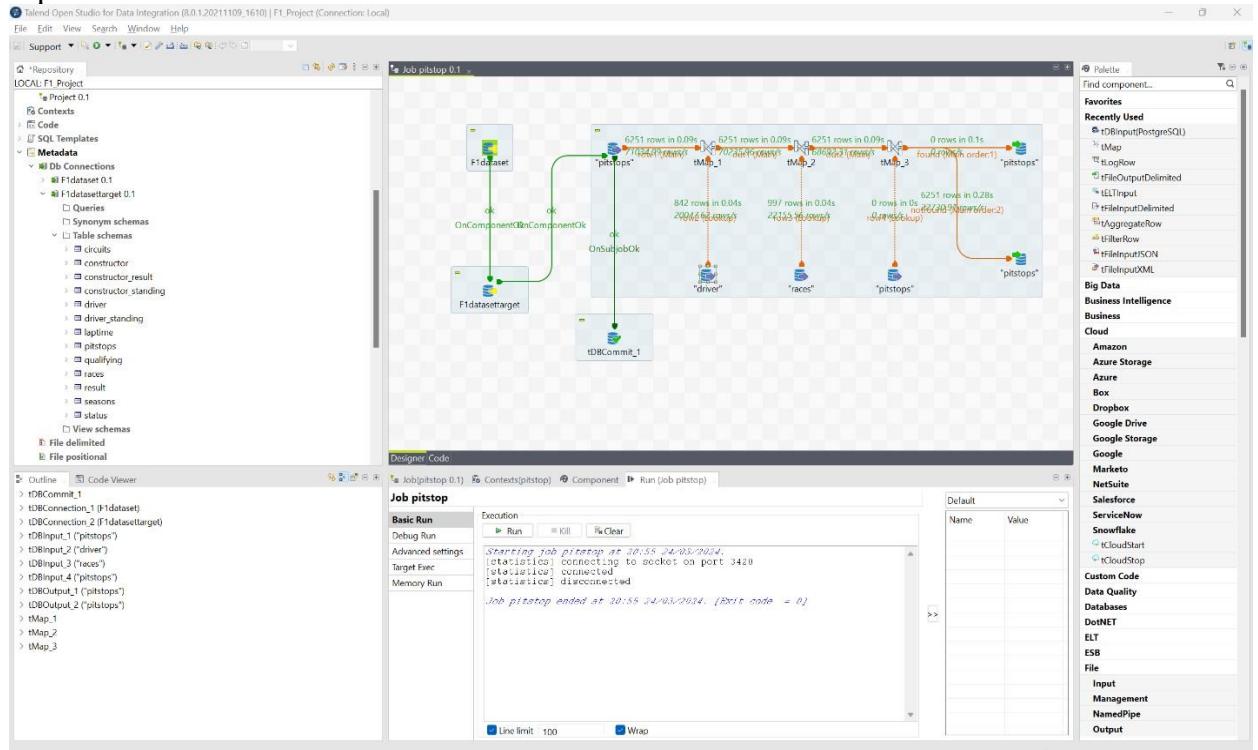
Data Output Messages Notifications

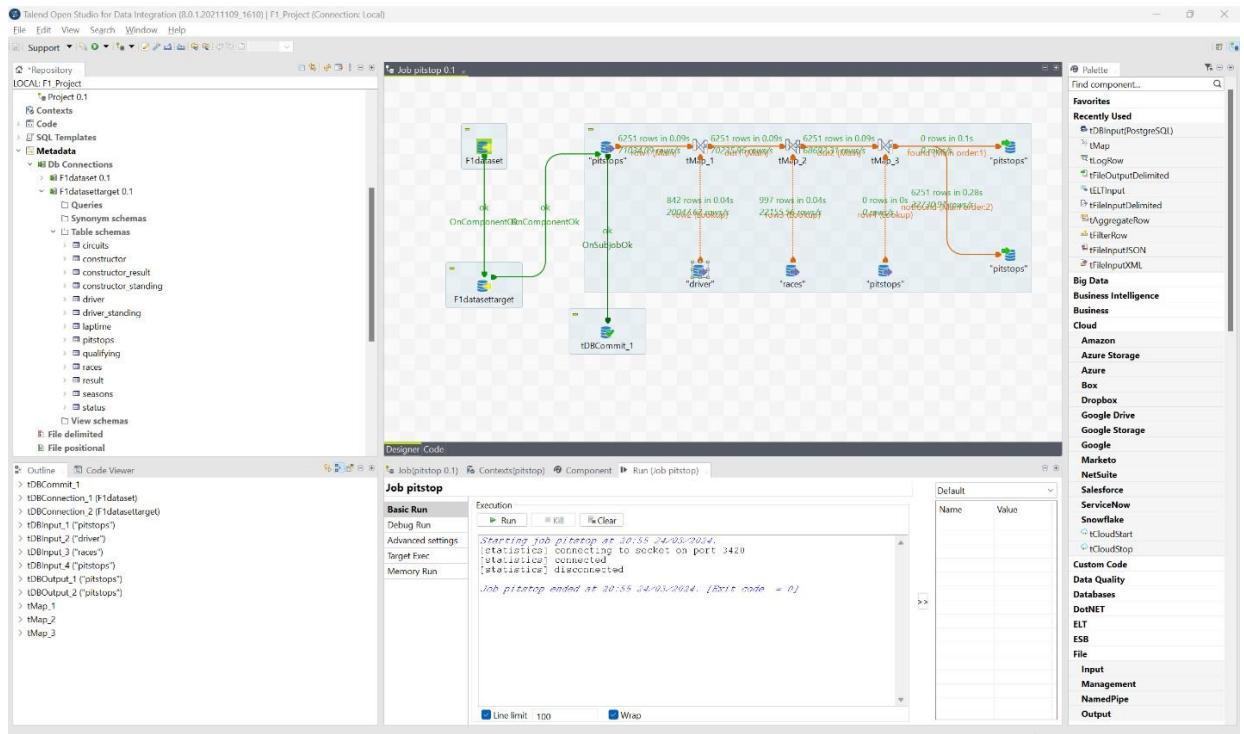
The screenshot shows a table titled "lap time dimension output" with the following columns and data:

	laptimkey [PK] integer	racekey integer	driverkey integer	lap_number integer	position integer	time character varying	milli_sec integer
1	1	840	20	1	1	1:38.109	98109
2	2	840	20	2	1	1:33.006	93006
3	3	840	20	3	1	1:32.713	92713
4	4	840	20	4	1	1:32.803	92803
5	5	840	20	5	1	1:32.342	92342
6	6	840	20	6	1	1:32.605	92605
7	7	840	20	7	1	1:32.502	92502
8	8	840	20	8	1	1:32.537	92537
9	9	840	20	9	1	1:33.240	93240
10	10	840	20	10	1	1:32.572	92572
11	11	840	20	11	1	1:32.669	92669
12	12	840	20	12	1	1:32.902	92902

Total rows: 1000 of 426633 Query complete 00:00:00.590 Ln 1, Col 22

12. Pitstop Dimension



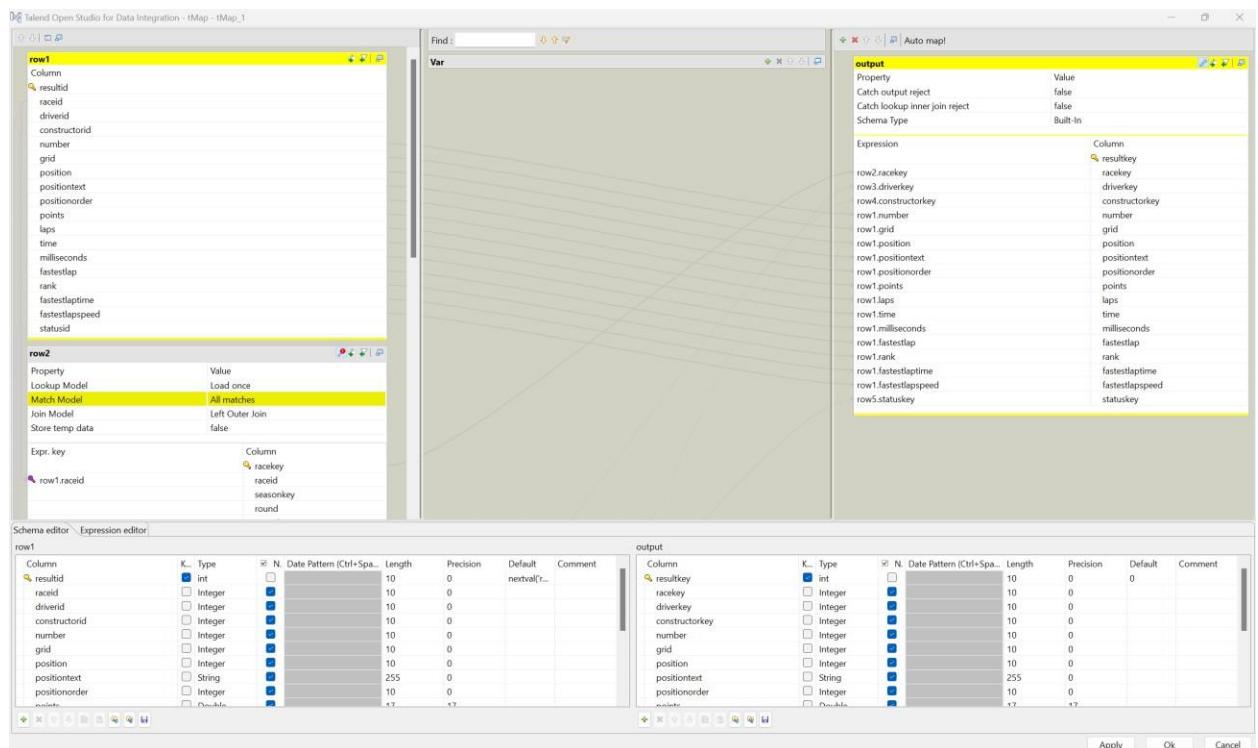
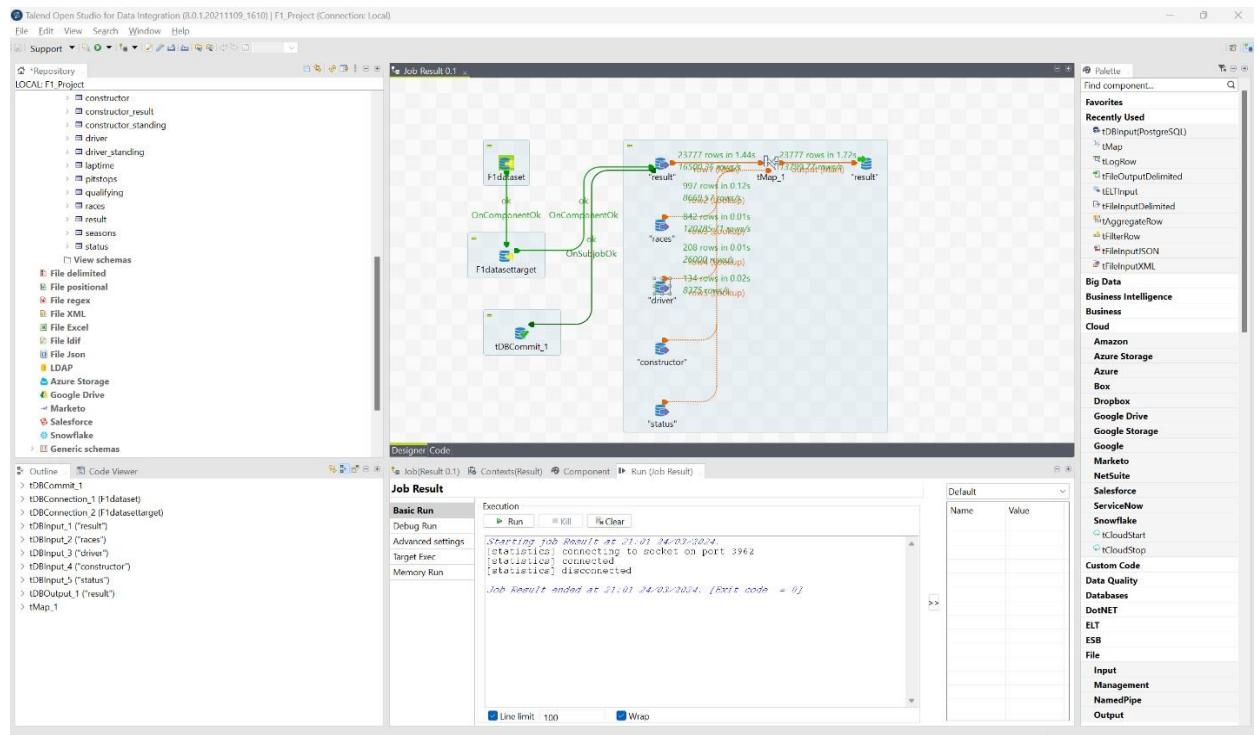


Pitstop Dimension Output

Data Output Messages Notifications

	pitstopkey [PK] integer	racekey integer	driverkey integer	stop integer	lap_number integer	time character varying (250)	milli_sec integer
1		1	840	153	1	1 17:05:23	26898
2		2	840	30	1	1 17:05:52	25021
3		3	840	17	1	11 17:20:48	23426
4		4	840	4	1	12 17:22:34	23251
5		5	840	13	1	13 17:24:10	23842
6		6	840	22	1	13 17:24:29	23643
7		7	840	20	1	14 17:25:17	22603
8		8	840	814	1	14 17:26:03	24863
9		9	840	816	1	14 17:26:50	25259
10		10	840	67	1	15 17:27:34	25342
11		11	840	2	1	15 17:27:41	22994
12		12	840	1	1	16 17:28:24	23227

13. Result Fact Table



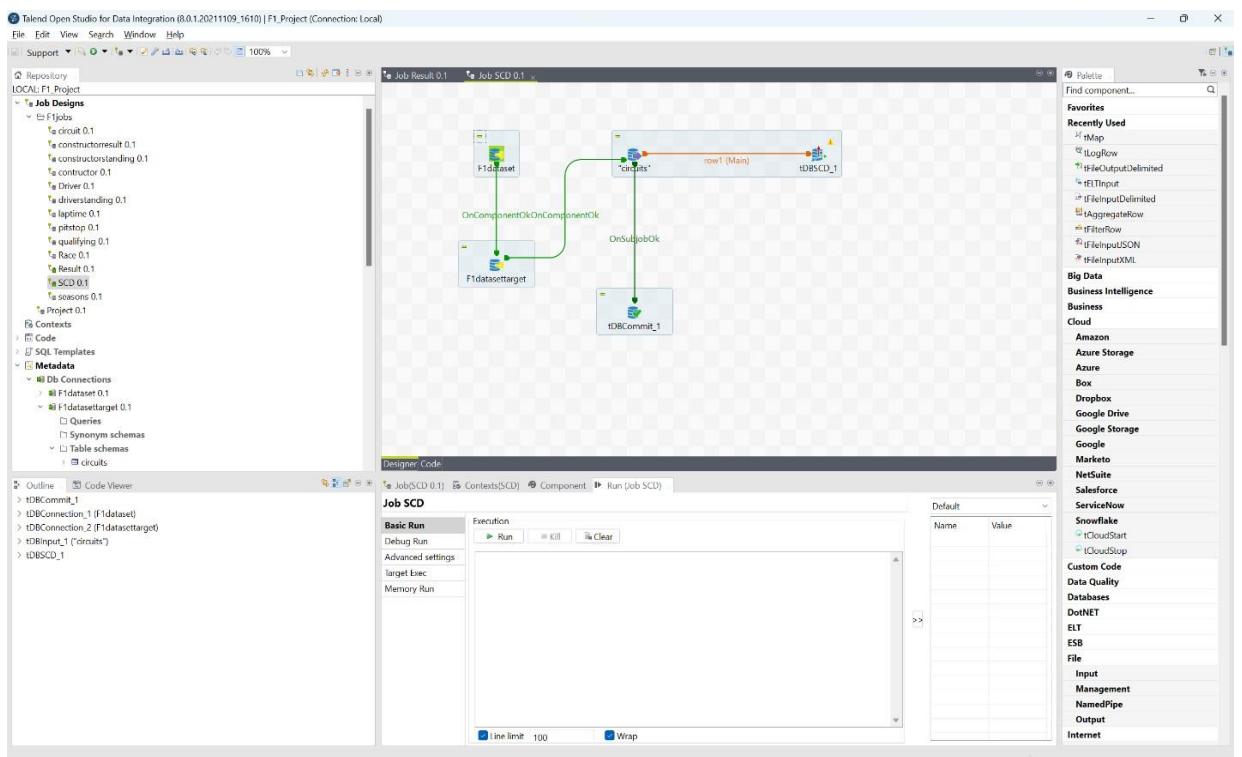
Result Fact Table Output

Data Output													
	resultkey [PK] integer	racekey integer	driverkey integer	constructorkey integer	number integer	grid integer	position integer	positiontext character varying (255)	positionorder integer	points double precision	laps integer		
1	1	18	1		1	22	1	1	1	10	5		
2	2	18	2		2	3	5	2	2	8	5		
3	3	18	3		3	7	7	3	3	6	5		
4	4	18	4		4	5	11	4	4	5	5		
5	5	18	5		1	23	3	5	5	4	5		
6	6	18	6		3	8	13	6	6	3	5		
7	7	18	7		5	14	17	7	7	2	5		
8	8	18	8		6	1	15	8	8	1	5		
9	9	18	9		2	4	2	[null]	R	0	4		
10	10	18	10		7	12	18	[null]	R	0	4		
11	11	18	11		8	18	19	[null]	R	0	3		
12	12	18	12		4	6	20	[null]	R	0	3		

Total rows: 1000 of 23777 Query complete 00:00:00.236

Ln 1, Col 21

14. SCD



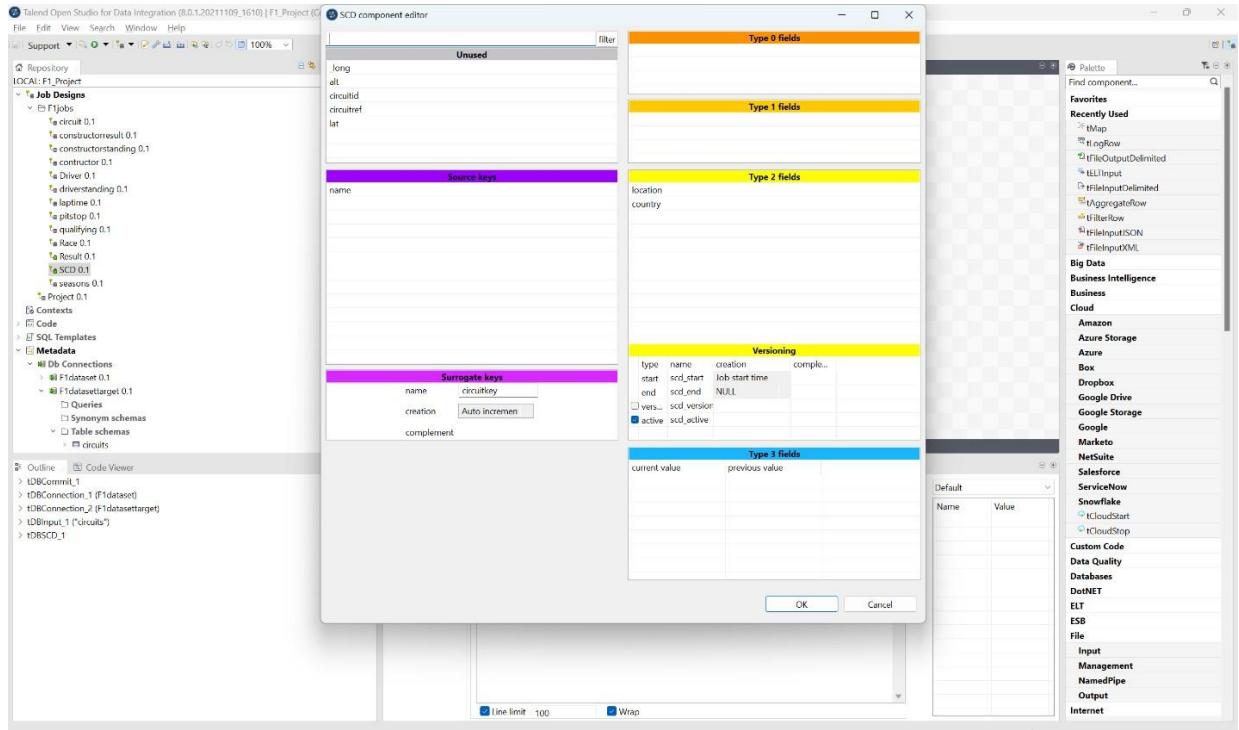
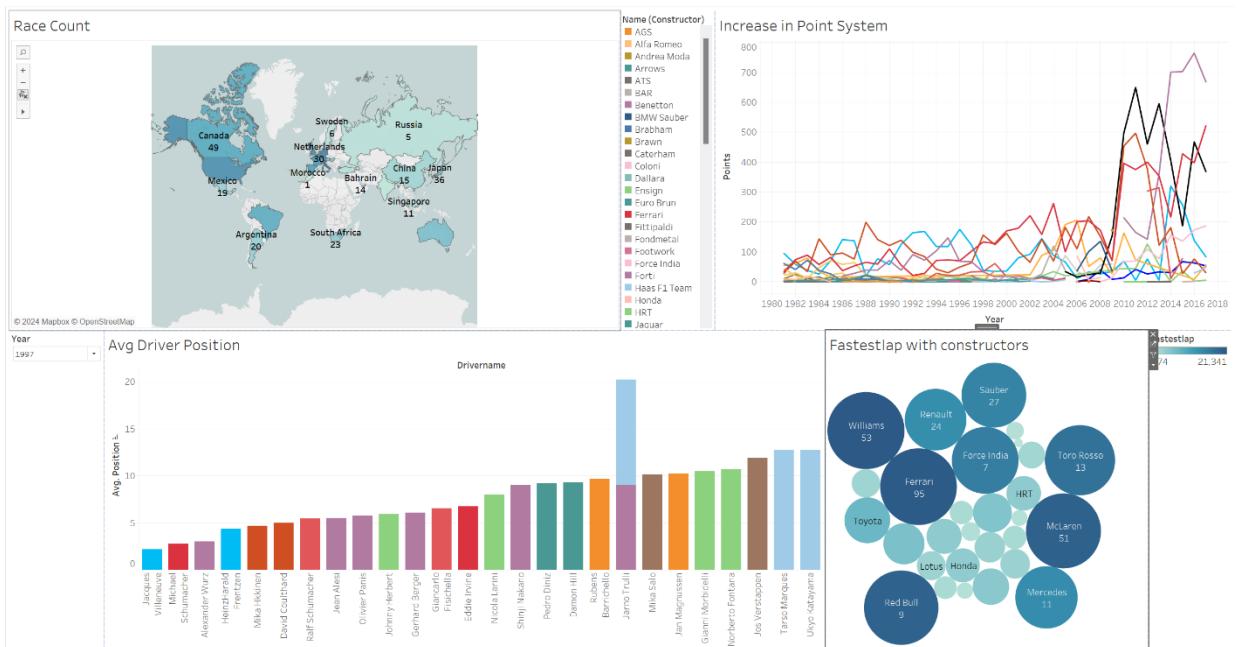


Tableau Dashboard



Airbnb Price Analysis

Project Proposal Summary:

We propose to transform the Boston Airbnb dataset, consisting of 95 columns, into dimension and fact tables. This structured approach will involve grouping related columns into dimension tables such as Listing, Host, Location, Property, Availability, and Review, while aggregating numerical data into a Fact Table capturing pricing details. By implementing this transformation, we aim to streamline data analysis, enhance decision-making processes, and provide valuable insights for Airbnb hosts and guests.

Analysis Objectives:

In addition to structuring the dataset into dimension and fact tables, we aim to conduct in-depth analysis on property listings based on geographical areas, pricing dynamics, and neighbourhood-specific trends. This analysis will involve:

Geographical Analysis:

- Segmenting property listings by geographical areas such as neighbourhoods, cities, and states.
- Analysing the distribution of listings across different areas to identify popular and emerging markets.
- Investigating spatial patterns and correlations between listing characteristics and geographical features.

Price Analysis:

- Studying the pricing variations across different geographical areas and neighbourhoods.
- Identifying factors influencing pricing decisions, such as property type, amenities, and seasonal demand.
- Conducting comparative analysis to determine price competitiveness and value propositions for listings in specific areas.

Neighbourhood Analysis:

- Examining property listings within individual neighbourhoods to understand localized demand and preferences.
- Assessing neighbourhood-specific amenities, attractions, and accessibility factors influencing listing popularity and pricing.
- Utilizing sentiment analysis or reviews data to gauge neighbourhood reputation and guest satisfaction levels.

Extra Charges Analysis:

- Investigating additional charges such as cleaning fees, security deposits, and fees for extra guests.
- Analysing the impact of extra charges on booking patterns, guest satisfaction, and overall listing performance.
- Identifying optimal pricing strategies for extra charges to maximize revenue while maintaining competitiveness.

Here's a description of each dimension and fact table along with their columns that we will make from raw dataset:

1. Listing Dimension Table:

- Listing_id: Unique identifier for each listing.
- Name: Name of the listing.
- Description: Description of the listing.
- Transit: Information about transportation options nearby.
- House_rules: Rules set by the host for guests.

- Interaction: Information about interaction with guests provided by the host.
- Guest_included: Number of guests included in the listing.
- Min_night: Minimum number of nights required for booking.
- Max_night: Maximum number of nights allowed for booking.
- Require_license: Indicates if a license is required for booking.
- Instant_bookable: Indicates if the listing supports instant booking.
- Cancellation_policy: Policy regarding cancellation of bookings.

2. Host Dimension Table:

- Id: Unique identifier for each host.
- Name: Name of the host.
- Host_since: Date when the host joined Airbnb.
- Host_location: Location of the host.
- Host_response_time: Response time of the host to inquiries.
- Host_response_rate: Rate of response by the host.
- Host_total_listings: Total number of listings managed by the host.
- Host_identity_verified: Indicates if the host's identity is verified.

3. Location Dimension Table:

- Street: Street address of the listing.
- Neighbourhood: Neighbourhood where the listing is located.
- City: City where the listing is located.
- State: State where the listing is located.
- Country: Country where the listing is located.
- Latitude: Latitude coordinates of the listing.
- Longitude: Longitude coordinates of the listing.
- Is_location_exact: Indicates if the location is exact.

4. Property Dimension Table:

- Property_type: Type of property (e.g., apartment, house, etc.).
- Room_type: Type of room (e.g., entire home, private room, shared room).
- Accommodates: Maximum number of guests the property can accommodate.
- Bathrooms: Number of bathrooms in the property.
- Bedrooms: Number of bedrooms in the property.
- Beds: Number of beds in the property.
- Amenities: Amenities provided in the property.

5. Availability Dimension Table:

- Availability_30: Number of days the property is available for booking in the next 30 days.
- Availability_60: Number of days the property is available for booking in the next 60 days.
- Availability_90: Number of days the property is available for booking in the next 90 days.
- Availability_365: Number of days the property is available for booking in the next 365 days.
- Calendar_updated: Date when the calendar was last updated.

6. Review Dimension Table:

- Number_of_reviews: Number of reviews received for the listing.
- Review_score_rating: Overall rating score given by guests.
- Cleanliness_score: Score given for cleanliness.
- Checking_score: Score given for the check-in process.
- Communication_score: Score given for communication with the host.
- Location_score: Score given for the location of the listing.
- Value_score: Score given for the value provided by the listing.

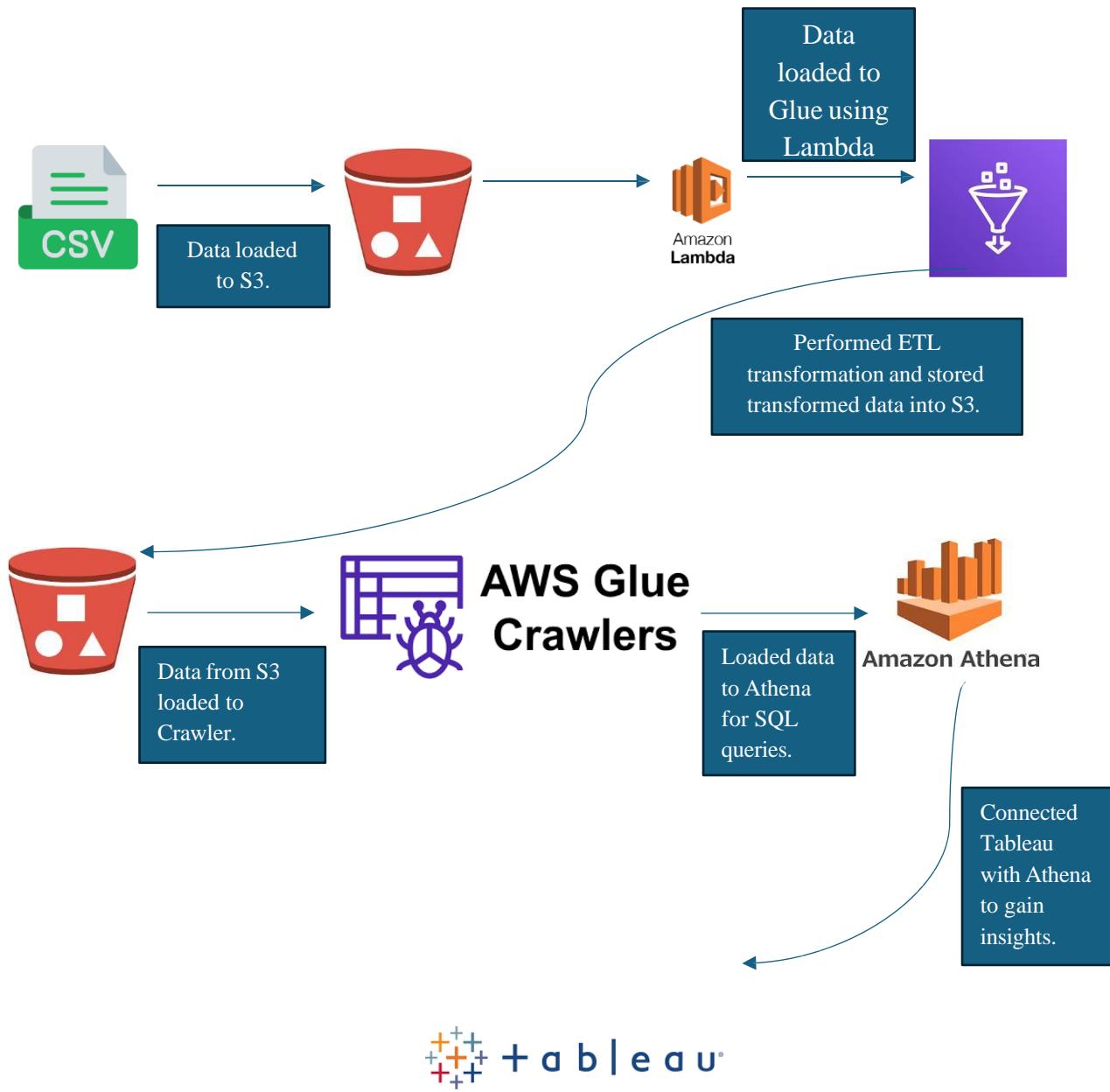
7. Fact Table:

- Price: Price per night for the listing.
- Weekly_price: Price for a week's stay.
- Security_deposit: Amount of security deposit required.
- Cleaning_fees: Fees for cleaning the property.
- Extra_people_price: Price for additional guests beyond the included number.

Citation

<https://www.kaggle.com/datasets/airbnb/boston>

Data Architecture



Data Ingestion

- Prepared structured data in CSV format.
- Created an S3 bucket, load csv file to S3.
- Used AWS lambda to fetch data from S3 and pass it to AWS glue.
- Used AWS glue for ETL to transform data.
- After transformation, sent data to the new S3 bucket.
- Loaded the transformed data to AWS Athena by AWS Glue Crawler.
- Used Athena to run queries and get granular insights from the data.

- Connected Athena with Tableau to visually get some insights.

The screenshot shows the AWS S3 console interface. On the left, there's a sidebar with options like Buckets, Access Grants, Access Points, Object Lambda Access Points, Multi-Region Access Points, Batch Operations, IAM Access Analyzer for S3, Block Public Access settings, Storage Lens, Dashboards, Storage Lens groups, AWS Organizations settings, Feature spotlight, and AWS Marketplace for S3. The main area displays an 'Account snapshot' with metrics: Total storage (8.6 MB), Object count (65), and Average object size (135.0 KB). A button 'View Storage Lens dashboard' is available. Below this is a table titled 'General purpose buckets (4)' with columns for Name, AWS Region, IAM Access Analyzer, and Creation date. The buckets listed are:

Name	AWS Region	IAM Access Analyzer	Creation date
aws-glue-assets-557425267574-us-east-1	US East (N. Virginia) us-east-1	View analyzer for us-east-1	April 3, 2024, 00:54:21 (UTC-04:00)
bostonairbnb-athena	US East (N. Virginia) us-east-1	View analyzer for us-east-1	April 5, 2024, 17:44:24 (UTC-04:00)
bostonrawcatabucket	US East (N. Virginia) us-east-1	View analyzer for us-east-1	April 1, 2024, 19:29:36 (UTC-04:00)
transformedairbnbdta	US East (N. Virginia) us-east-1	View analyzer for us-east-1	April 3, 2024, 02:34:04 (UTC-04:00)

The screenshot shows the AWS Glue Studio script editor. The left sidebar has sections for Getting started, ETL jobs, Visual ETL, Notebooks, Job run monitoring, Data Catalog tables, Data connections, Workflows (orchestration), Data Catalog, Databases, Tables, Stream schema registries, Schemas, Connections, Crawlers, Classifiers, Catalog settings, Data Integration and ETL, Legacy pages, What's New, Documentation, AWS Marketplace, Enable compact mode, and Enable new navigation. The main area is titled 'lambda_invoking_etl_pipeline' and shows a 'Script' tab with the following Python code:

```

1 import sys
2 from awsglue.transforms import *
3 from awsglue.utils import getResolvedOptions
4 from pyspark.context import SparkContext
5 from awsglue.context import GlueContext
6 from awsglue.job import Job
7 import gs_sequence_id
8 import gs_mil_rnwk
9
10 args = getResolvedOptions(sys.argv, ['JOB_NAME', 'S3_SOURCE_PATH', 'S3_DEST_PATH'])
11 sc = SparkContext()
12 glueContext = GlueContext(sc)
13 spark = glueContext.spark_session
14 job = Job(glueContext)
15 job.init(args['JOB_NAME'], args)
16
17 # Script generated for node Amazon S3
18 AmazonS3_node1712121756462 = glueContext.create_dynamic_frame.from_options(format_options={"quoteChar": "\'", "withHeader": True, "separator": ","}, optimizePerformance": False, connection_type="s3", format="csv", connection_options={"path": [args['S3_SOURCE_PATH']]}, "recurse": True, transformation_ctx="AmazonS3_node1712121756462")
19
20 # Script generated for node Identifier
21 Identifier_node17121217572769 = AmazonS3_node1712121756462.gs_sequence_id(colName="id")
22
23 # Script generated for node Remove NULL Rows
24 RemoveNullRows_node1712125720207 = Identifier_node1712123732769.gs_null_rows()
25
26 # Script generated for node Change Schema
27 ChangeSchema_node1712183779730 = ApplyMapping.apply(frame=RemoveNullRows_node1712125720207, mappings=[{"id": "bigint", "ic": "long"}, {"listing_url": "string"}])
28
29 ChangeSchema_node1712183779730 = ApplyMapping.apply(frame=RemoveNullRows_node1712125720207, mappings=[{"id": "bigint", "ic": "long"}, {"listing_url": "string"}])

```

The code uses AWS Glue context and transformations to process data from an S3 source, generate sequence IDs, remove null rows, and change schema. The script is saved with a timestamp of 4/3/2024, 10:45:20 PM.

Query editor | Athena | us-east-1

Tables (1) Views (0)

transformedairbnbdata

SQL Ln 1, Col 1

Run again Explain Cancel Clear Create

Reuse query results up to 60 minutes ago

Completed Time in queue: 74 ms Run time: 1.012 sec Data scanned: 336.09 KB

Results (3,585)

#	id	listing_url	scrape_id	last_scraped	name	experiences_of
1	0	https://www.airbnb.com/rooms/12147973	2.02E+13	9/7/2016	Sunny Bungalow in the City	none
2	1	https://www.airbnb.com/rooms/3075044	2.02E+13	9/7/2016	Charming room in pet friendly apt	none
3	2	https://www.airbnb.com/rooms/6976	2.02E+13	9/7/2016	Mexican Folk Art Haven in Boston	none
4	3	https://www.airbnb.com/rooms/1436513	2.02E+13	9/7/2016	Spacious Sunny Bedroom Suite in Historic Home	none
5	4	https://www.airbnb.com/rooms/7651065	2.02E+13	9/7/2016	Come Home to Boston	none
6	5	https://www.airbnb.com/rooms/12386020	2.02E+13	9/7/2016	Private Bedroom + Great Coffee	none
7	6	https://www.airbnb.com/rooms/5706985	2.02E+13	9/7/2016	New Lrg Studio apt 15 min to Boston	none
8	7	https://www.airbnb.com/rooms/2843445	2.02E+13	9/7/2016	Tranquility on Top of the Hill	none
9	8	https://www.airbnb.com/rooms/753446	2.02E+13	9/7/2016	6 miles away from downtown Boston!	none

Tableau Dashboard

