Adolescent Alcohol Abuse Analysis – Oklahoma BRFSS Data Warehouse

Sruthi Kondra

Email: kondra.s@northeastern.edu

College of Professional Studies, Northeastern University

Introduction:

This project focuses on designing a data warehouse for analyzing adolescent alcohol consumption trends in Oklahoma using data from the Behavioral Risk Factor Surveillance System (BRFSS). The primary objective is to structure and optimize the dataset by implementing a star schema model, ensuring efficient data retrieval and analysis. By constructing a well-defined dimensional model, this project aims to enhance analytical capabilities, enabling targeted insights into high-risk groups, geographic patterns, and demographic influences on adolescent alcohol abuse.

The dataset selected for this analysis comprises survey responses collected through the BRFSS, specifically filtered for alcohol consumption-related questions in Oklahoma. This dataset includes both qualitative and quantitative attributes, such as demographic information, survey questions, participant responses, and geographic details like city and county. Given its complexity, the dataset requires careful normalization to reduce redundancy and improve query performance. To achieve this, dimension tables were created for locations, survey questions, responses, demographic breakouts, and breakout categories, while a central fact table stores the core survey data.

To build and analyze the data warehouse, I utilized MySQL for database management and SQL queries for data extraction and transformation. MySQL Workbench was used for schema design, data modeling, and visualization through an Entity-Relationship Diagram (ERD). Additionally, the data was preprocessed and imported using structured SQL commands, ensuring data integrity and consistency. The final analysis involved executing complex queries to extract insights related to adolescent alcohol consumption by age group, city, and county, with the goal of identifying high-risk populations and informing data-driven public health strategies.

Data Exploration & Creation of Initial Star Schema

After conducting an in-depth exploration of the Behavioral Risk Factor Surveillance System (BRFSS) data, specifically for Oklahoma, I developed a strong understanding of the dataset's structure and its key components related to adolescent alcohol consumption. The dataset provided a wealth of both qualitative and quantitative data, offering deep insights into various risk behaviors, demographic characteristics, and health-related trends across the state. A crucial part of this dataset was the demographic information, which was distributed across two separate files: one exclusively covering Oklahoma-specific demographics and another containing broader U.S. demographic data.

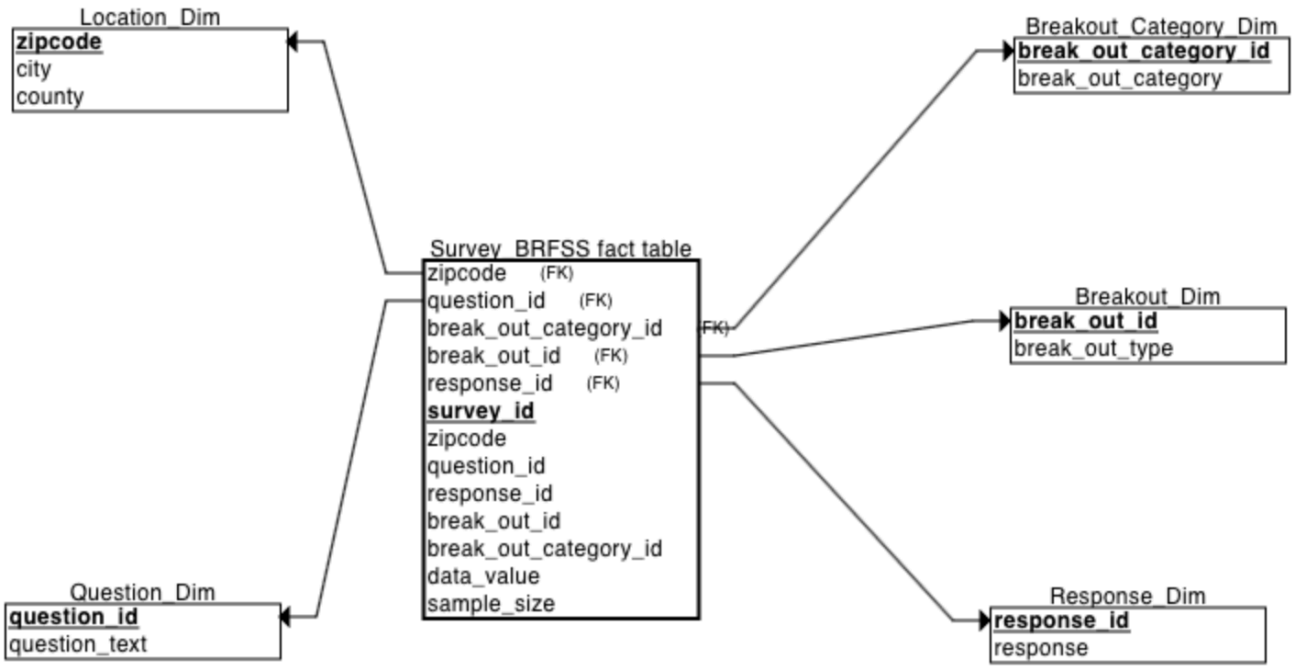
Upon initial examination, I identified that the survey fact table contained a mix of categorical and numerical data, requiring careful normalization to improve data processing and query efficiency. To address this, I

segregated qualitative attributes (such as questions, responses, and demographic categories) into dedicated dimension tables, ensuring that the fact table focused solely on measurable metrics. Additionally, I consolidated the demographic details from the two different datasets into a single location dimension table, reducing redundancy while ensuring a unified and structured approach to demographic analysis.

Using ERDPlus, I constructed a Star Schema to transform this dataset into a structured data warehouse model, making it more optimized for analytical querying. At the core of this schema is the Survey BRFSS Fact Table, which stores key survey metrics, such as the data value (number of affirmative responses to alcohol-related questions) and sample size. Surrounding the fact table, multiple dimension tables provide meaningful context and allow for efficient data retrieval:

- Location Dimension – Stores geographic data such as zip code, city, and county, enabling spatial analysis of survey responses.
- Question Dimension – Contains the survey questions, allowing analysis based on specific topics related to alcohol consumption.
- Response Dimension – Captures the different response types (e.g., Yes, No) to facilitate behavioral analysis.
- Breakout Dimension – Segments data into demographic groups such as age, gender, income level, or education attainment to support targeted analysis.
- Breakout Category Dimension – Groups demographic breakouts into broader categories (e.g., "Age Group", "Income Level"), ensuring a structured classification system.

The Star Schema was chosen because it significantly enhances query performance and data retrieval efficiency, making it particularly effective for large datasets such as the BRFSS survey. By implementing this Star Schema, I have established a structured and scalable data model that allows for efficient analysis of adolescent alcohol consumption trends in Oklahoma. This model provides a solid foundation for SQL-based querying, enabling the identification of high-risk age groups, geographic hotspots, and demographic disparities. The next steps will involve leveraging this schema to execute complex SQL queries, extracting insights that can inform public health initiatives and policy decisions aimed at addressing adolescent alcohol abuse.



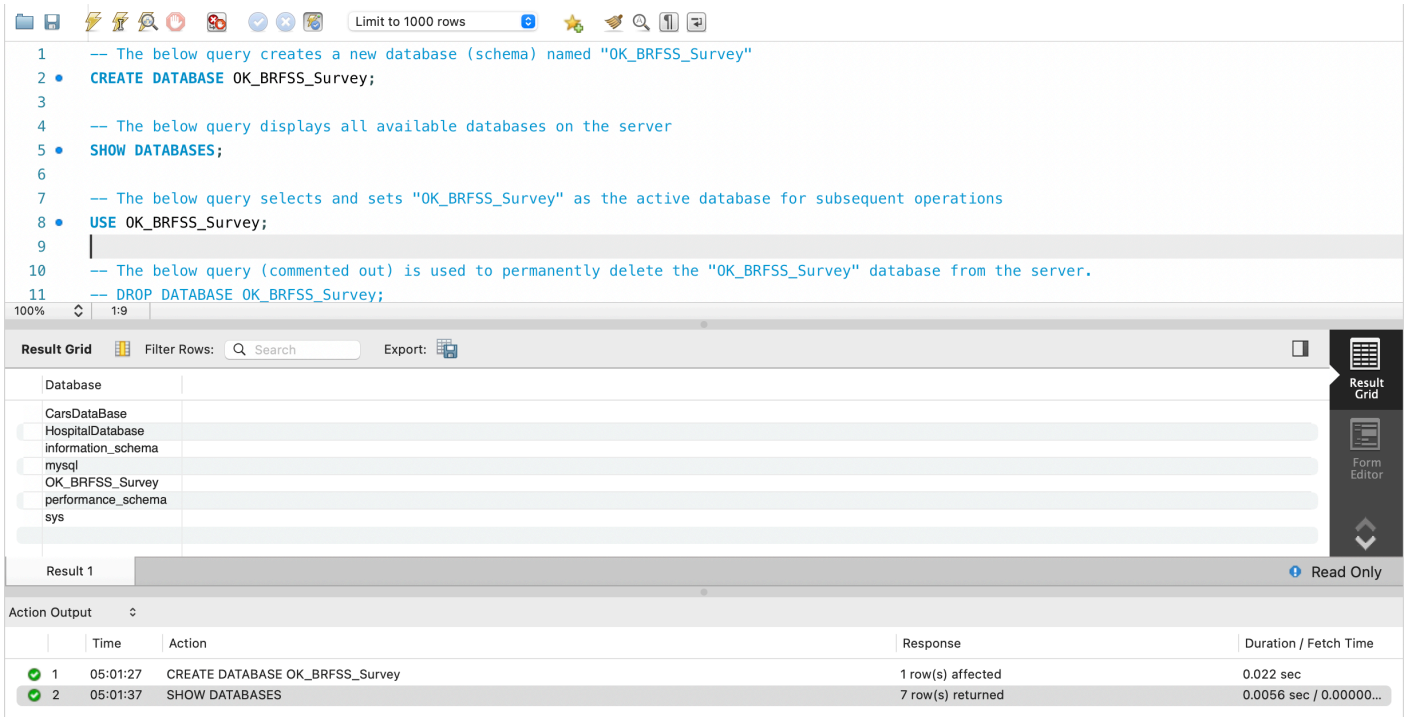
Analysis

Creating the Database and Importing Data

1. OK_BRFSS_Survey Table

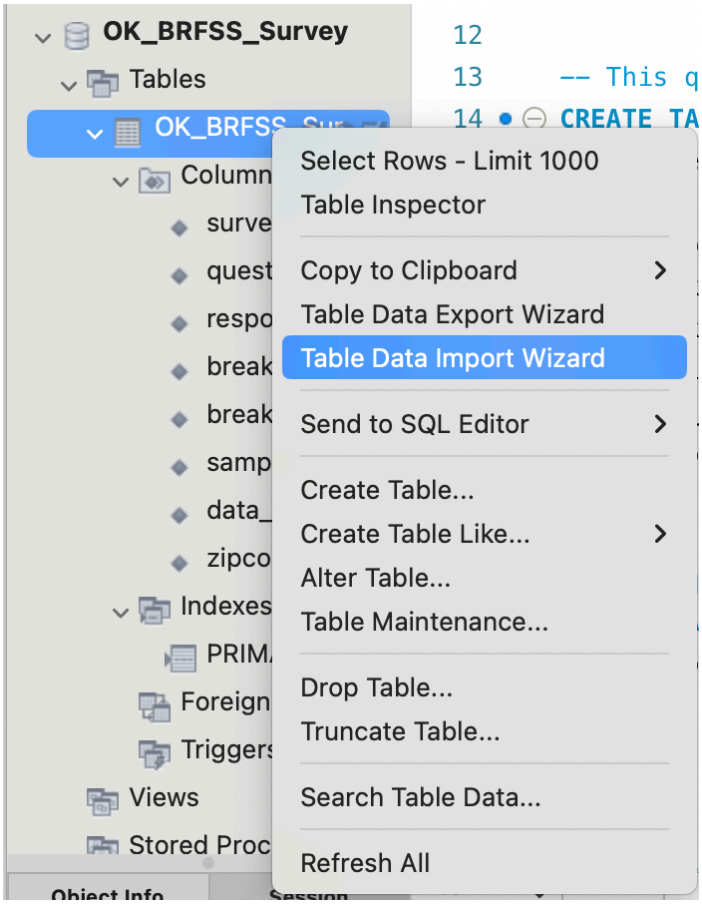
To begin the analysis, I first created a dedicated database schema named OK_BRFSS_Survey to store and manage the survey data effectively. This database was designed to hold information from the Behavioral Risk Factor Surveillance System (BRFSS), specifically focusing on alcohol consumption-related survey responses in Oklahoma. The creation of this database ensures structured data management and efficient querying for subsequent analysis.

Once the database was created, I proceeded with defining the necessary tables. The OK_BRFSS_Survey table was structured to capture key details such as survey questions, responses, demographic categories, and the corresponding zip codes of respondents. This table serves as the foundation for the fact and dimension tables in the star schema. The SQL script used to create the table includes constraints like primary keys, foreign keys, and data validation rules to maintain data integrity. A screenshot of the table creation process is provided below



After setting up the database and table, the next step involved importing the survey dataset into MySQL. The **Table Data Import Wizard** was used to efficiently load the CSV file containing the survey responses. The import process involved selecting the CSV file, mapping its columns to the corresponding table fields, and verifying the data before committing it to the database. Screenshots detailing the import process, including file selection, mapping, and the successful completion of the import, are included below

```sql
-- This query creates a staging table for initial survey data related to alcohol consumption in Oklahoma.
CREATE TABLE OK_BRFSS_Survey (
    survey_id INT PRIMARY KEY AUTO_INCREMENT, -- Unique identifier for each survey response (Auto-incrementing primary key)
    question_text VARCHAR(255), -- The survey question asked (Limited to 255 characters)
    response VARCHAR(30) CHECK (response IN ('Yes', 'No')), -- Response to the question (Only "Yes" or "No" allowed)
    break_out VARCHAR(100), -- General category of respondent demographics (e.g., "Household Income")
    break_out_category VARCHAR(100), -- Specific demographic category (e.g., "$15,000-$24,999")
    sample_size INT, -- Total number of respondents who answered the survey question
    data_value INT, -- Number of respondents who answered "Yes"
    zipcode VARCHAR(10) -- Zip code of the respondent
);
```


OK_BRFSS_Survey
Tables
Column
surve
quest
respo
break
break
samp
data_
zipco
Indexes
PRIM
Foreign
Triggers
Views
Stored Proc
Object Info
Session
12
13 -- This q
14 CREATE TA
Select Rows - Limit 1000
Table Inspector
Copy to Clipboard
Table Data Export Wizard
Table Data Import Wizard
Send to SQL Editor
Create Table...
Create Table Like...
Alter Table...
Table Maintenance...
Drop Table...
Truncate Table...
Search Table Data...
Refresh All

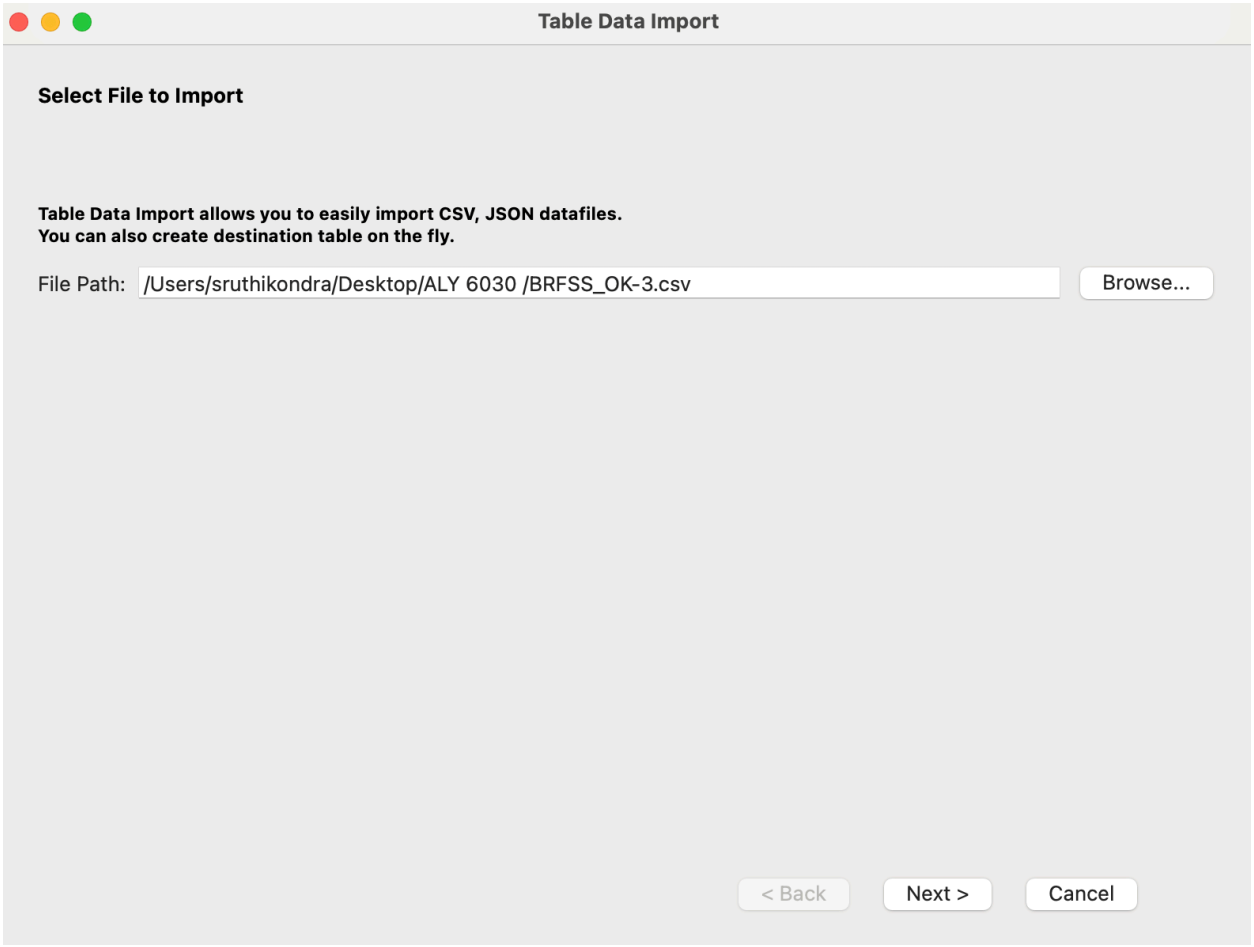


Table Data Import
Select File to Import
Table Data Import allows you to easily import CSV, JSON datafiles.
You can also create destination table on the fly.
File Path: /Users/sruthikondra/Desktop/ALY 6030 /BRFSS_OK-3.csv
Browse...
< Back
Next >
Cancel

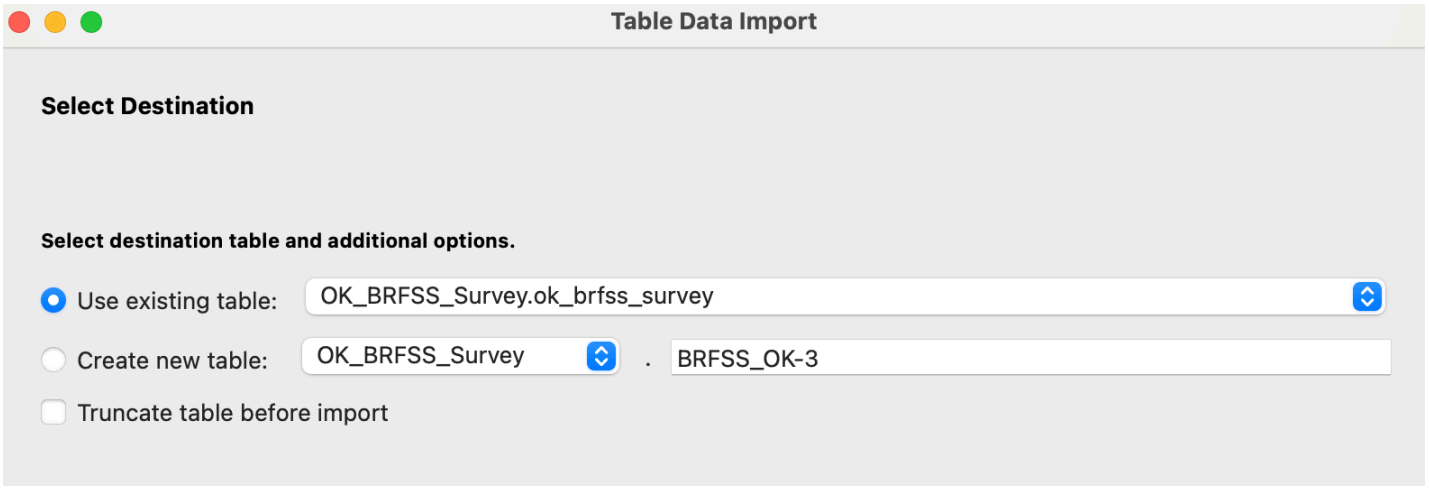


Table Data Import
Select Destination
Select destination table and additional options.
Use existing table:
OK_BRFSS_Survey.ok_brfss_survey
Create new table:
OK_BRFSS_Survey
.
BRFSS_OK-3
Truncate table before import

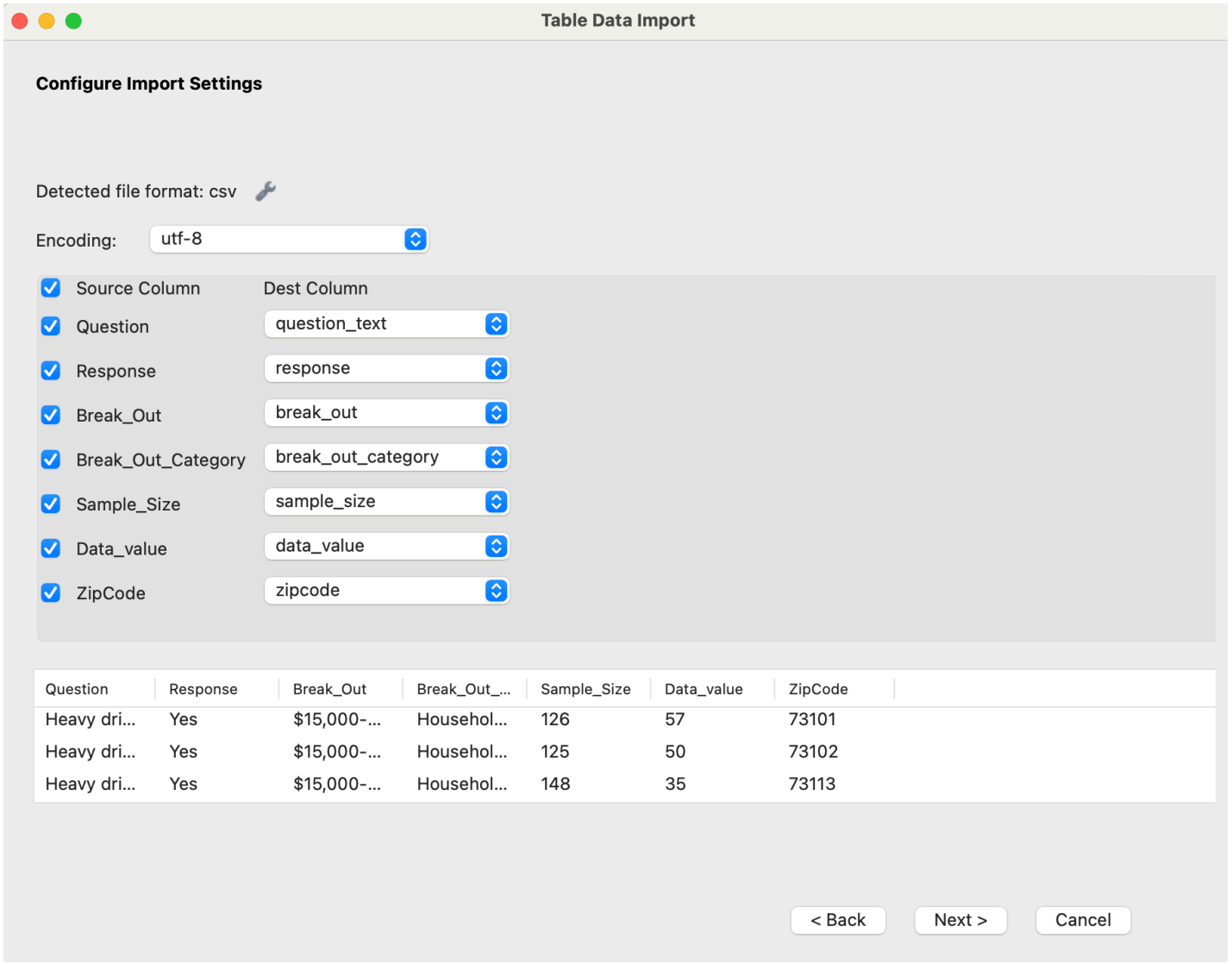


Table Data Import
Configure Import Settings
Detected file format: csv
Encoding:
utf-8
Source Column
Dest Column
Question
question_text
Response
response
Break_Out
break_out
Break_Out_Category
break_out_category
Sample_Size
sample_size
Data_value
data_value
ZipCode
zipcode
Question
Response
Break_Out
Break_Out_...
Sample_Size
Data_value
ZipCode
Heavy dri...
Yes
$15,000-...
Househol...
126
57
73101
Heavy dri...
Yes
$15,000-...
Househol...
125
50
73102
Heavy dri...
Yes
$15,000-...
Househol...
148
35
73113
< Back
Next >
Cancel

Import Data

The following tasks will now be performed. Please monitor the execution.

- Prepare Import
- Import data file

Finished performing tasks. Click [Next >] to continue.

Import Results

File /Users/sruthikondra/Downloads/BRFSS_OK-3.csv was imported in 0.719 s

Table OK_BRFSS_Survey.ok_brfss_survey has been used

667 records imported

To confirm that the data was successfully imported, I executed a simple SELECT query to display the first few records from the table. The result confirmed that all 667 records were loaded accurately, ensuring that the database was ready for further transformation and analysis.

```
25    -- Retrieve all records from the 'OK_BRFSS_Survey' table to verify that data has been imported successfully.
26 •  SELECT * FROM OK_BRFSS_Survey; -- Displays the contents of the table.
27
```

100% 3:23

Result Grid | Filter Rows: Search | Edit: | Export/Import:

survey_id	question_text	response	break_out	break_out_categ...	sample_size	data_value	zipcode
1	Heavy drinkers (adult men having more than tw...	Yes	$15,000-$24,999	Household Income	126	57	73101
2	Heavy drinkers (adult men having more than tw...	Yes	$15,000-$24,999	Household Income	125	50	73102
3	Heavy drinkers (adult men having more than tw...	Yes	$15,000-$24,999	Household Income	148	35	73113
4	Heavy drinkers (adult men having more than tw...	Yes	$15,000-$24,999	Household Income	145	89	73123
5	Heavy drinkers (adult men having more than tw...	Yes	$15,000-$24,999	Household Income	159	42	73124
6	Heavy drinkers (adult men having more than tw...	Yes	$15,000-$24,999	Household Income	113	38	73125
7	Heavy drinkers (adult men having more than tw...	Yes	$15,000-$24,999	Household Income	110	92	73126
8	Heavy drinkers (adult men having more than tw...	Yes	$15,000-$24,999	Household Income	121	62	73136
9	Heavy drinkers (adult men having more than tw...	Yes	$15,000-$24,999	Household Income	140	29	73140
10	Heavy drinkers (adult men having more than tw...	Yes	$15,000-$24,999	Household Income	181	86	73143

With the database successfully populated, the next phase involved constructing a star schema to optimize data retrieval and facilitate more structured analysis of adolescent alcohol consumption trends in Oklahoma.

Demographics_OK Table

Next, I created a dedicated Demographics_OK table to store location-based information, including ZIP codes, cities, and counties specific to Oklahoma. This step was crucial for linking survey responses to geographic areas, enabling deeper insights into regional patterns in adolescent alcohol abuse. The Demographics_OK table was structured with three key fields: zipcode as the primary key, city, and county, ensuring that each ZIP code uniquely identifies a location.

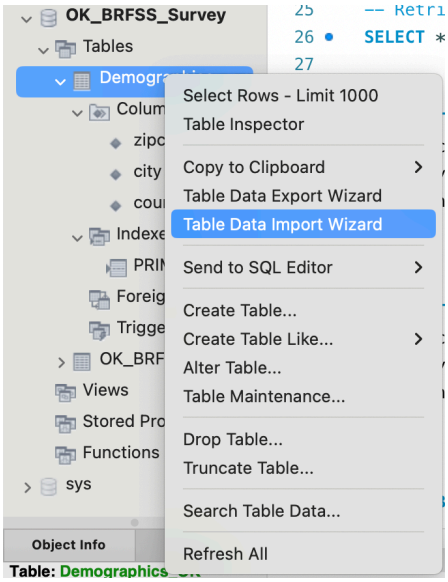
The SQL script for creating this table is shown in Figure below where the schema is defined to store demographic details efficiently. This step was followed by verifying that the table was successfully created in the database.

```sql
-- This query creates a staging table for demographic data specific to Oklahoma, including zip code, city, and county.
CREATE TABLE Demographics_OK (
    zipcode VARCHAR(10) PRIMARY KEY, -- Unique zip code as primary key
    city VARCHAR(100), -- City corresponding to the zip code
    county VARCHAR(100) -- County corresponding to the zip code
);
```

Importing Data into the Demographics Table

After creating the table, I imported location-specific data from a CSV file containing ZIP codes, city names, and county information. Using MySQL's Table Data Import Wizard, I selected the dataset file, mapped the columns correctly to match the table schema, and initiated the import process. This process is illustrated in screenshots below where I selected the appropriate file, configured import settings, and verified the successful insertion of data into the table.

Once the import was complete, I executed a SELECT query to retrieve all records from the Demographics_OK table, confirming that 764 records were successfully loaded.



Select File to Import

Table Data Import allows you to easily import CSV, JSON datafiles.
You can also create destination table on the fly.

File Path: /Users/sruthikondra/Downloads/Demographics_OK-1.csv Browse...

ALY6030_Assignment_1_SruthiKondra

< Back Next > Cancel

Select Destination

Select destination table and additional options.

Use existing table: OK_BRFSS_Survey.demographics_ok

Create new table: OK_BRFSS_Survey . Demographics_OK-1

Truncate table before import

< Back Next > Cancel

Configure Import Settings

Detected file format: csv

Encoding: utf-8

Source Column	Dest Column
ZipCode	zipcode
City	city
County	county

Sprintax Nonresident Tax Session - Northeastern University.ics

ZipCode	City	County
73071	NORMAN	CLEVELA...
73072	NORMAN	CLEVELA...
73085	YUKON	CANADIAN

< Back | Next > | Cancel

Import Data

The following tasks will now be performed. Please monitor the execution.

- Prepare Import
- Import data file

Finished performing tasks. Click [Next >] to continue.

Import Results

File /Users/sruthikondra/Downloads/Demographics_OK-1.csv was imported in 0.538 s

Table OK_BRFSS_Survey.demographics_ok has been used

764 records imported

The output of this query is below, displaying the imported ZIP codes, city names, and county data.

```
35    -- Retrieve all records from the 'Demographics_OK' table to verify that data has been imported successfully
36 •  SELECT * FROM Demographics_OK; -- Displays the contents of the table.
37
```

100% 31:36

Result Grid Filter Rows: Search Edit: Export/Import:

zipcode	city	county
73001	ALBERT	CADDO
73002	ALEX	GRADY
73003	EDMOND	OKLAHOMA
73004	AMBER	GRADY
73005	ANADARKO	CADDO
73006	APACHE	CADDO
73007	ARCADIA	OKLAHOMA
73008	BETHANY	OKLAHOMA
73009	BINGER	CADDO
73010	BLANCHARD	MCCLAIN
73011	BRADLEY	GRADY
73012	EDMOND	OKLAHOMA
73013	EDMOND	OKLAHOMA
73014	CALUMET	CANADIAN
73015	CARNEGIE	CADDO

By structuring the demographic data into a separate dimension table, I ensured seamless integration with the fact table, allowing for efficient querying and analysis when examining alcohol abuse trends across different geographic areas in Oklahoma.

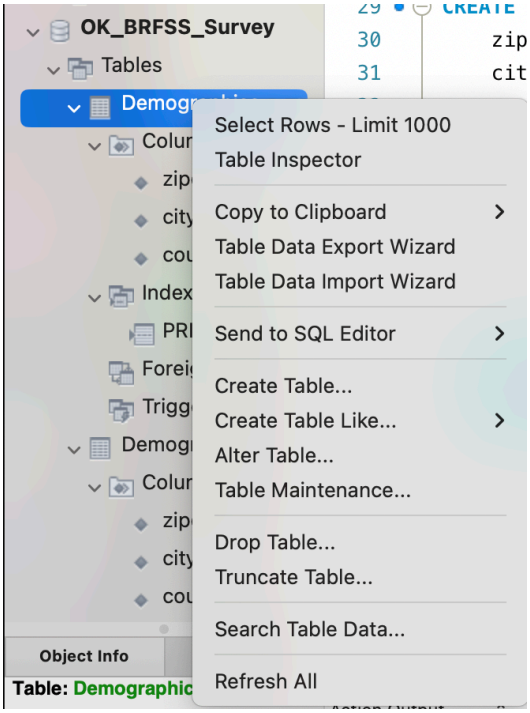
Demographics Table

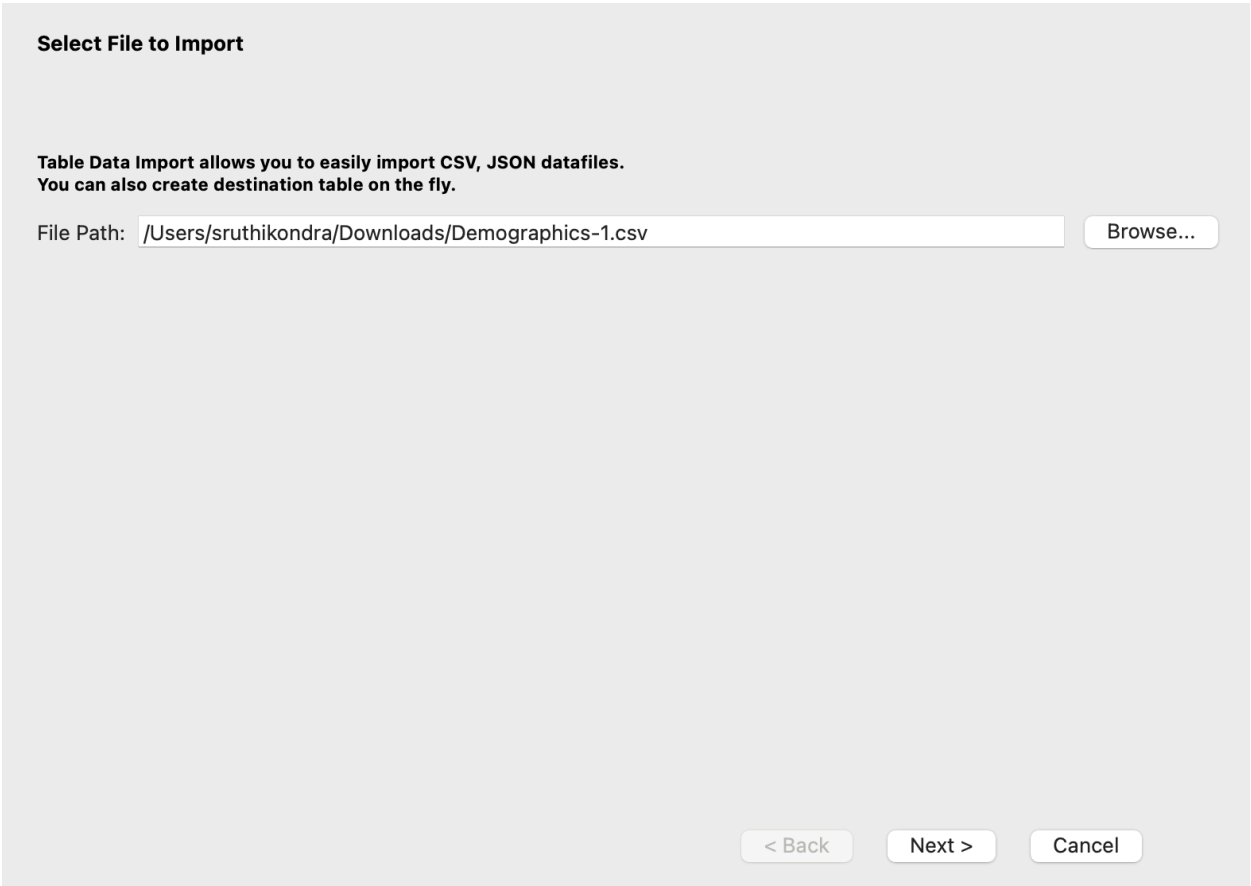
Next I created a Demographics table. This table serves as a staging area for storing location-related details such as zip code, city, and county, which are crucial for analyzing survey responses at a regional level. The Demographics table was designed to cover all U.S. locations, ensuring that the dataset remains comprehensive and scalable.

The CREATE TABLE statement defines the schema, where the zipcode column is the primary key since it uniquely identifies each location. The city and county fields provide additional geographic context, enabling further breakdowns in the analysis.

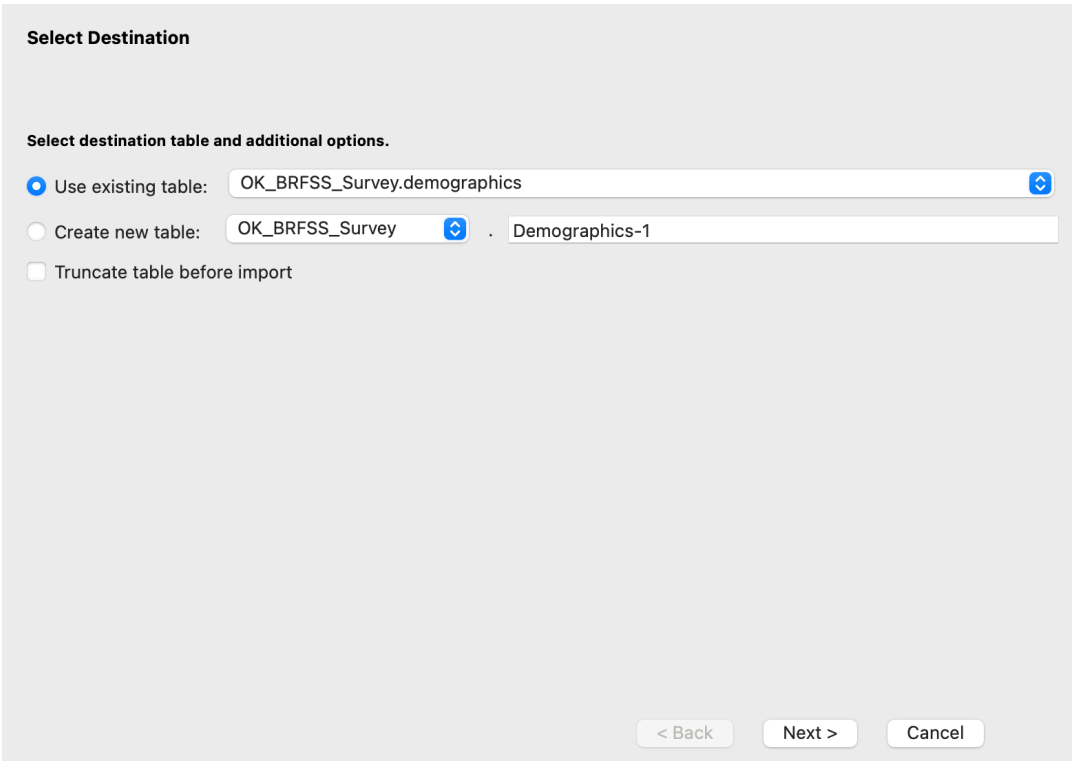
```
-- This query creates a staging table for general demographic data covering all US locations, including zip code, city, and county.
CREATE TABLE Demographics (
    zipcode VARCHAR(10) PRIMARY KEY, -- Unique zip code as primary key
    city VARCHAR(100), -- City corresponding to the zip code
    county VARCHAR(100) -- County corresponding to the zip code
);
```

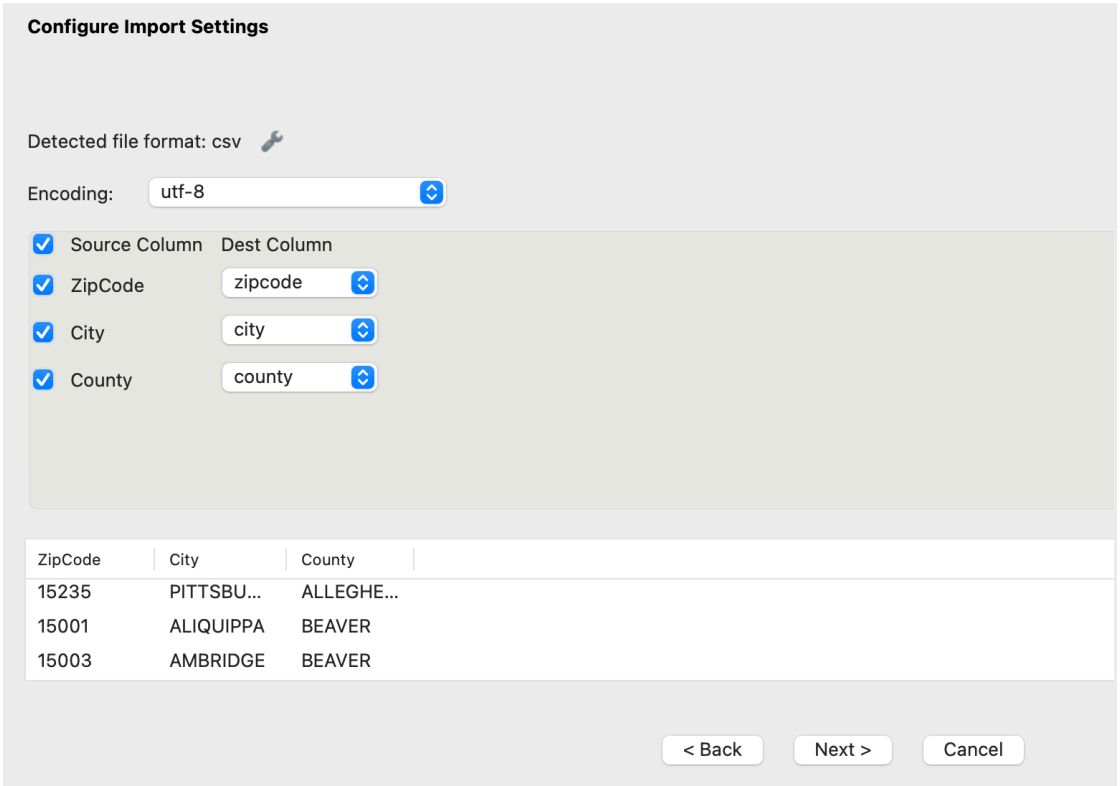
Once the table structure was in place, the next step was to load the data. This was done using the Table Data Import Wizard, which allows for an easy and structured approach to importing large datasets in CSV format. The import process starts by selecting the appropriate table in the database and then specifying the CSV file containing the demographic data





After selecting the file, I mapped the source columns from the CSV to their corresponding destination columns in the database The zipcode, city, and county fields were correctly mapped to ensure data consistency. Once the mapping was verified, I proceeded with the import process, which executed successfully, as indicated below





Import Data

The following tasks will now be performed. Please monitor the execution.

- Prepare Import
- Import data file

Finished performing tasks. Click [Next >] to continue.

Import Results

File /Users/sruthikondra/Downloads/Demographics-1.csv was imported in 1.365 s

Table OK_BRFSS_Survey.demographics has been used

2185 records imported

Finally, I ran a SELECT query to retrieve all records from the Demographics table to confirm that the data had been successfully imported and stored correctly. The result grid displaying the imported records is shown below. This validation step was essential to ensure that the imported data was accurate and ready for integration with the main fact table.

```
45    -- Retrieve all records from the 'Demographics' table to verify that data has been imported successfully.
46    SELECT * FROM Demographics; -- Displays the contents of the table.
47
```

100% 67:46

Result Grid | Filter Rows: Search | Edit: | Export/Import: | Fetch rows:

zipcode	city	county
15001	ALIQUIPPA	BEAVER
15003	AMBRIDGE	BEAVER
15004	ATLASBURG	WASHINGTON
15005	BADEN	BEAVER
15006	BAIRDFORD	ALLEGHENY
15007	BAKERSTOWN	ALLEGHENY
15009	BEAVER	BEAVER
15010	BEAVER FALLS	BEAVER
15012	BELLE VERNON	FAYETTE
15014	BRACKENRIDGE	ALLEGHENY

Creating Dimension Tables

Question Dimension Table:

The question_dim table was created as a dimension table to store survey questions separately, ensuring efficient data organization and eliminating redundancy. This table contains two key columns: question_id, which serves as an auto-incremented primary key, and question_text, a VARCHAR(255) field that stores the actual text of the survey questions. By creating a separate table for questions, we optimize database performance by reducing text repetition and ensuring that each unique question is stored only once.

Separating questions into a dedicated dimension table is essential for maintaining third normal form (3NF), which prevents data duplication, enhances consistency, and makes querying more efficient. Instead of storing lengthy question texts repeatedly in the main fact table, we reference them using a unique question_id. This approach simplifies data retrieval while improving database performance, especially when working with large datasets.

After creating the table, I populated it by inserting distinct questions from the OK_BRFSS_Survey dataset using a SELECT DISTINCT query. This step ensures that no duplicate questions exist in the table. A SELECT query was then run to verify successful data insertion, confirming that each question was captured correctly. The output showed that only one unique question was present, "Heavy drinkers (adult men having more than two drinks per day and adult women having more than one drink per day)", validating that the dataset contains a single survey question. Additionally, there were no null values in the question column, indicating that data integrity was preserved.

```
51    -- Creating the Question Dimension Table
52    -- This table stores unique survey questions with an auto-increment primary key.
53    CREATE TABLE Question_Dim (
54        question_id INT PRIMARY KEY AUTO_INCREMENT, -- Unique ID for each question
55        question_text VARCHAR(255) -- Stores the survey question text
56    );
57
58    -- Insert distinct questions from the OK_BRFSS_Survey table into Question_Dim
59    INSERT INTO Question_Dim (question_text)
60    SELECT DISTINCT question_text FROM OK_BRFSS_Survey;
61
62    -- Display all entries in the Question_Dim table to verify data insertion
63    SELECT * FROM Question_Dim;
64
```

100% 74:62

Result Grid Filter Rows: Search Edit: Export/Import:

question_id	question_text
1	Heavy drinkers (adult men having more than tw…
NULL	NULL

Response Dimension Table:

The response_dim table was introduced to store unique survey responses efficiently, ensuring proper data normalization and integrity. This table consists of two key columns: response_id, an auto-incremented primary key, and response, a VARCHAR(30) field that captures survey responses. Since survey participants can typically answer only "Yes" or "No," this structure is ideal for storing responses in a normalized format.

By creating this table, we eliminate redundancy from the main survey dataset. Instead of repeating text responses across multiple records, we reference each response using a unique response_id. This design aligns with third normal form (3NF) principles, improving data management and ensuring consistency. Moreover, the use of a primary key enforces uniqueness, preventing duplicate responses from being stored.

To populate this table, I used a SELECT DISTINCT query to extract unique responses from the OK_BRFSS_Survey dataset. This ensures that only distinct responses— "Yes" and "No"—are stored. Running a SELECT query confirmed that the insertion was successful, showing that both possible response values were correctly captured. The structure allows for efficient querying and facilitates the linking of responses with survey questions in the fact table.

```
66    -- Creating the Response Dimension Table
67    -- This table stores survey response options ("Yes" or "No") with an auto-increment primary key.
68    CREATE TABLE Response_Dim (
69        response_id INT PRIMARY KEY AUTO_INCREMENT, -- Unique ID for each response
70        response VARCHAR(30) CHECK (response IN ('Yes', 'No')) -- Stores response values
71    );
72
73    -- Insert distinct responses from OK_BRFSS_Survey into Response_Dim
74    INSERT INTO Response_Dim (response)
75    SELECT DISTINCT response FROM OK_BRFSS_Survey;
76
77    -- Display all entries in the Response_Dim table to verify data insertion
78    SELECT * FROM Response_Dim;
79
80
```

Copy of Copy of Agreed to be Contacted CBBB Applicants (Edit Access).xlsx

100% 1:79

Result Grid Filter Rows: Search Edit: Export/Import:

response_id	response
1	Yes
NULL	NULL

Location Dimension Table:

The location_dim table was designed to store geographic information, including zip codes, cities, and counties, to facilitate structured analysis of survey responses based on location. This table contains three columns: zipcode, which serves as the primary key, city, and county. The use of a separate table for location details

ensures that geographic data is standardized and eliminates redundant storage of city and county names in the main survey dataset.

In real-world datasets, location-based analysis is crucial for understanding trends across different regions. Instead of repeatedly storing the same city and county names in every survey record, we reference them through a unique zipcode. This approach improves database efficiency, reduces storage requirements, and ensures data consistency.

To populate this table, I extracted distinct zipcode, city, and county values from two existing tables, Demographics_OK and Demographics. A UNION query was used to merge data from both sources while maintaining only unique records. This method ensures that duplicate locations are eliminated. A COUNT(*) query was run to verify the total number of inserted records, confirming that the expected number of unique locations was successfully captured. The location dimension table provides a structured way to filter and analyze survey responses based on geographic areas.

```
81    -- Creating the Location Dimension Table
82    -- This table stores geographic details like city and county, using zip code as the primary key.
83    CREATE TABLE Location_Dim (
84        zipcode VARCHAR(10) PRIMARY KEY, -- Unique identifier for location
85        city VARCHAR(100), -- City associated with the zip code
86        county VARCHAR(100) -- County associated with the zip code
87    );
88
89    -- Insert distinct location data from both Demographics_OK and Demographics into Location_Dim
90    INSERT INTO Location_Dim (zipcode, city, county)
91    SELECT DISTINCT zipcode, city, county FROM (
92        SELECT zipcode, city, county FROM Demographics_OK
93        UNION
94        SELECT zipcode, city, county FROM Demographics
95    ) AS combined_locations;
96
97    -- Display count of all entries in Location_Dim to confirm data insertion
98    SELECT COUNT(*) FROM Location_Dim;
```

100% 25:95

Result Grid Filter Rows: Search Export:

COUNT(*)
2949

Breakout Dimension Table:

The breakout_dim table was created to categorize survey respondents based on demographic attributes such as income levels, age groups, and other classifications. This table consists of two key columns: break_out_id, an auto-incremented primary key, and break_out_type, a VARCHAR(100) field that defines demographic categories.

In a survey dataset, respondents often belong to different demographic groups, such as age brackets, income levels, or educational backgrounds. Storing these classifications in a separate table reduces redundancy in the

main survey dataset while allowing for efficient grouping and filtering of responses. Instead of storing textual descriptions of demographic groups repeatedly, we reference them using a unique break_out_id.

To ensure only distinct breakout categories were stored, I used a SELECT DISTINCT query to extract unique values from the OK_BRFSS_Survey dataset. Running a SELECT query confirmed that all expected categories, such as "$15,000-$24,999," "25-34," "College graduate," and "Male", were correctly captured. This structure supports efficient analysis of demographic trends within the dataset.

```sql
-- Creating the Breakout Dimension Table
-- This table categorizes respondents based on demographic groups like income or age.
CREATE TABLE Breakout_Dim (
    break_out_id INT PRIMARY KEY AUTO_INCREMENT, -- Unique ID for each breakout category
    break_out_type VARCHAR(100) -- Type of breakout (e.g., "Household Income", "Age Group")
);

-- Insert distinct breakout types from OK_BRFSS_Survey into Breakout_Dim
INSERT INTO Breakout_Dim (break_out_type)
SELECT DISTINCT break_out FROM OK_BRFSS_Survey;

-- Display all entries in the Breakout_Dim table to verify data insertion
SELECT * FROM Breakout_Dim;
```

```sql
-- Display all entries in the Breakout_Dim table to verify data insertion
SELECT * FROM Breakout_Dim;
```

100% 1:111

Result Grid Filter Rows: Search Edit: Export/Import:

break_out_id	break_out_type
1	$15,000-$24,999
2	$25,000-$34,999
3	$35,000-$49,999
4	$50,000+
5	18-24
6	25-34
7	35-44
8	45-54
9	55-64
10	65+
11	Black, non-Hisp…
12	College graduate
13	Female
14	H.S. or G.E.D.
15	Hispanic
16	Less than $15,…
17	Less than H.S.
18	Male
19	Multiracial, non-…
20	Other, non-Hisp…
21	Overall
22	Some post-H.S.
23	White, non-His…

Breakout Category Dimension Table:

The breakout_category_dim table was created to store more specific demographic classifications within the breakout types. This table includes two columns: break_out_category_id, an auto-incremented primary key, and break_out_category, a VARCHAR(100) field representing categories such as Household Income, Education Level, Age Group, Race/Ethnicity.

The purpose of this table is to provide a more granular breakdown of demographic classifications, improving the clarity and structure of the dataset. By normalizing these values into a separate dimension table, we ensure that demographic categories are well-defined and consistently referenced throughout the survey analysis.

A SELECT DISTINCT query was used to extract unique breakout categories from the OK_BRFSS_Survey dataset. The verification query confirmed that all expected values were correctly inserted, ensuring the completeness of demographic classifications in the dataset.

```
-- Creating the Breakout Category Dimension Table
-- This table stores specific demographic values within breakout types.
CREATE TABLE Breakout_Category_Dim (
    break_out_category_id INT PRIMARY KEY AUTO_INCREMENT, -- Unique ID for each breakout category
    break_out_category VARCHAR(100) -- Specific demographic category (e.g., "$15,000-$24,999")
);

-- Insert distinct breakout categories from OK_BRFSS_Survey into Breakout_Category_Dim
INSERT INTO Breakout_Category_Dim (break_out_category)
SELECT DISTINCT break_out_category FROM OK_BRFSS_Survey;

-- Display all entries in the Breakout_Category_Dim table to verify data insertion
SELECT * FROM Breakout_Category_Dim;
```

100% 47:131

Result Grid | Filter Rows: Search | Edit: | Export/Import:

break_out_category...	break_out_categ...
1	Household Income
2	Age Group
3	Race/Ethnicity
4	Education Attained
5	Gender
6	Overall
NULL	NULL

Fact Table Creation and Data Insertion

After setting up the necessary dimension tables, I created the BRFSS_Survey_Fact table, which serves as the central fact table in the star schema. This table stores survey responses and links them to the relevant dimension tables using foreign keys. The structure includes the survey_id, an auto-incrementing primary key, and several foreign keys referencing the Location_Dim, Question_Dim, Response_Dim, Breakout_Dim, and Breakout_Category_Dim tables. Additionally, it contains two numerical columns: sample_size, representing the number of respondents, and data_value, indicating the count of respondents who answered "Yes" to the survey question.

By designing the fact table this way, we ensure that each survey response is properly linked to its corresponding attributes, allowing for efficient querying and analysis. The normalization achieved here eliminates redundancy, optimizes storage, and provides a structured way to retrieve data based on locations, questions, responses, and demographic breakouts.

```
-- Creating the Fact Table (BRFSS_Survey_Fact)
-- This table stores the actual survey responses, linking to all dimension tables.
CREATE TABLE BRFSS_Survey_Fact (
    survey_id INT PRIMARY KEY AUTO_INCREMENT, -- Unique ID for each survey response
    zipcode VARCHAR(10), -- Foreign key linking to Location_Dim
    question_id INT, -- Foreign key linking to Question_Dim
    response_id INT, -- Foreign key linking to Response_Dim
    break_out_id INT, -- Foreign key linking to Breakout_Dim
    break_out_category_id INT, -- Foreign key linking to Breakout_Category_Dim
    sample_size INT, -- Total respondents for the survey question
    data_value INT, -- Number of respondents who answered "Yes"
    FOREIGN KEY (zipcode) REFERENCES Location_Dim (zipcode),
    FOREIGN KEY (question_id) REFERENCES Question_Dim (question_id),
    FOREIGN KEY (response_id) REFERENCES Response_Dim (response_id),
    FOREIGN KEY (break_out_id) REFERENCES Breakout_Dim (break_out_id),
    FOREIGN KEY (break_out_category_id) REFERENCES Breakout_Category_Dim (break_out_category_id)
);
```

Once the fact table was created, I populated it by extracting data from the staging table OK_BRFSS_Survey and joining it with the dimension tables. The INSERT INTO statement selects relevant attributes from OK_BRFSS_Survey, matches them with corresponding dimension tables using JOIN operations, and inserts them into BRFSS_Survey_Fact.

This step ensures that instead of storing raw data with repeating information, each row in the fact table is structured with dimension table references. It optimizes query performance by minimizing redundancy while maintaining data integrity. The join operation links survey responses with their respective locations, questions, response options, and demographic details.

After running the insertion query, I verified the data by selecting all entries from the fact table. The results confirmed that each survey response had been correctly mapped with appropriate dimension table references, maintaining consistency and integrity in the database.

```
-- Insert data into the Fact Table by joining staging and dimension tables
INSERT INTO BRFSS_Survey_Fact (zipcode, question_id, response_id, break_out_id, break_out_category_id, sample_size, data_value)
SELECT
    l.zipcode,
    q.question_id,
    r.response_id,
    bo.break_out_id,
    boc.break_out_category_id,
    bs.sample_size,
    bs.data_value
FROM
    OK_BRFSS_Survey bs
    JOIN Location_Dim l ON bs.zipcode = l.zipcode
    JOIN Question_Dim q ON bs.question_text = q.question_text
    JOIN Response_Dim r ON bs.response = r.response
    JOIN Breakout_Dim bo ON bs.break_out = bo.break_out_type
    JOIN Breakout_Category_Dim boc ON bs.break_out_category = boc.break_out_category;

-- Display all entries from the Fact Table to verify data insertion
SELECT * FROM BRFSS_Survey_Fact;
```

```
167    -- Display all entries from the Fact Table to verify data insertion
168 •  SELECT * FROM BRFSS_Survey_Fact;
169
```

100% 62:162

Result Grid Filter Rows: Search Edit: Export/Import:

survey_id	zipcode	question_id	response_id	break_out_id	break_out_category...	sample_size	data_value
1	73196	1	1	1	1	125	48
2	73195	1	1	1	1	190	11
3	73194	1	1	1	1	177	83
4	73190	1	1	1	1	151	78
5	73189	1	1	1	1	199	85
6	73185	1	1	1	1	187	87
7	73184	1	1	1	1	174	161
8	73178	1	1	1	1	115	81
9	73164	1	1	1	1	195	5
10	73157	1	1	1	1	141	121
11	73156	1	1	1	1	101	26
12	73155	1	1	1	1	128	88
13	73154	1	1	1	1	100	68
14	73153	1	1	1	1	166	137
15	73152	1	1	1	1	113	57
16	73148	1	1	1	1	148	26
17	73147	1	1	1	1	113	59
18	73146	1	1	1	1	163	97
19	73144	1	1	1	1	124	59
20	73143	1	1	1	1	181	86

Once the data was successfully transferred to the dimensional model, I dropped the staging tables (OK_BRFSS_Survey, Demographics_OK, and Demographics). These tables were initially used to load and preprocess the raw data before normalizing it into the fact and dimension tables. Since all necessary transformations and insertions were completed, the staging tables were no longer needed, and removing them helped declutter the database while preserving the structured dimensional model.

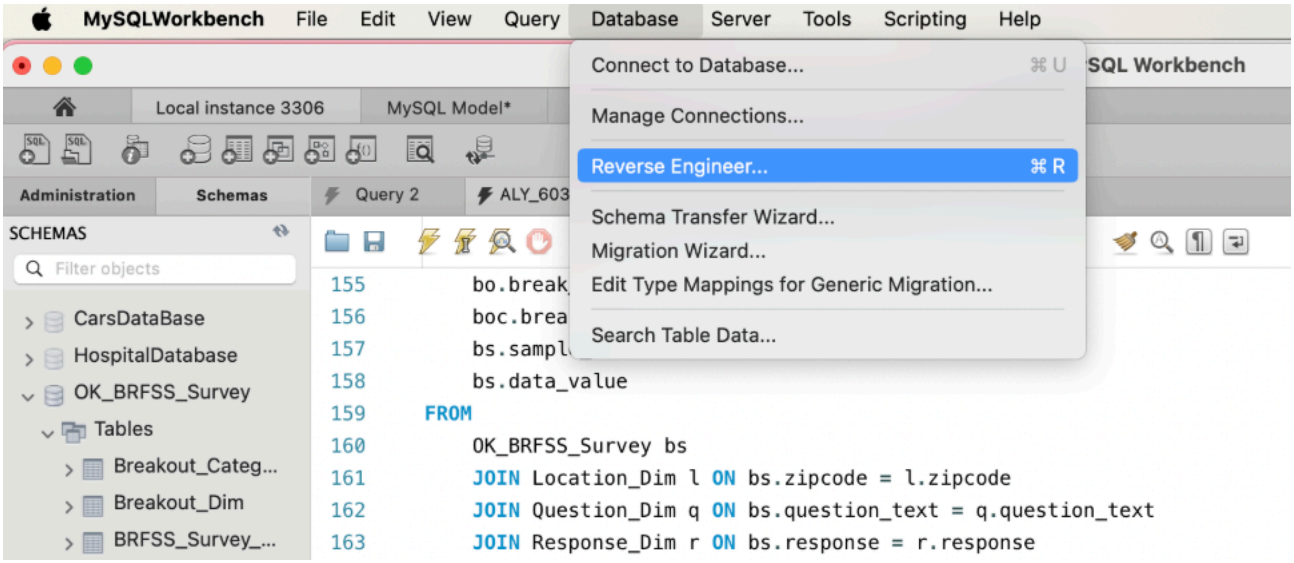
```
170    -- The below queries drop staging tables after data has been successfully transferred into the dimensional model
171 •  DROP TABLE OK_BRFSS_Survey;
172 •  DROP TABLE Demographics_OK;
173 •  DROP TABLE Demographics;
```

This final step ensures that the database is fully optimized for querying, with the fact table efficiently storing response data and linking it to well-structured dimension tables. This setup facilitates easy analysis of survey trends based on location, demographics, and response patterns.

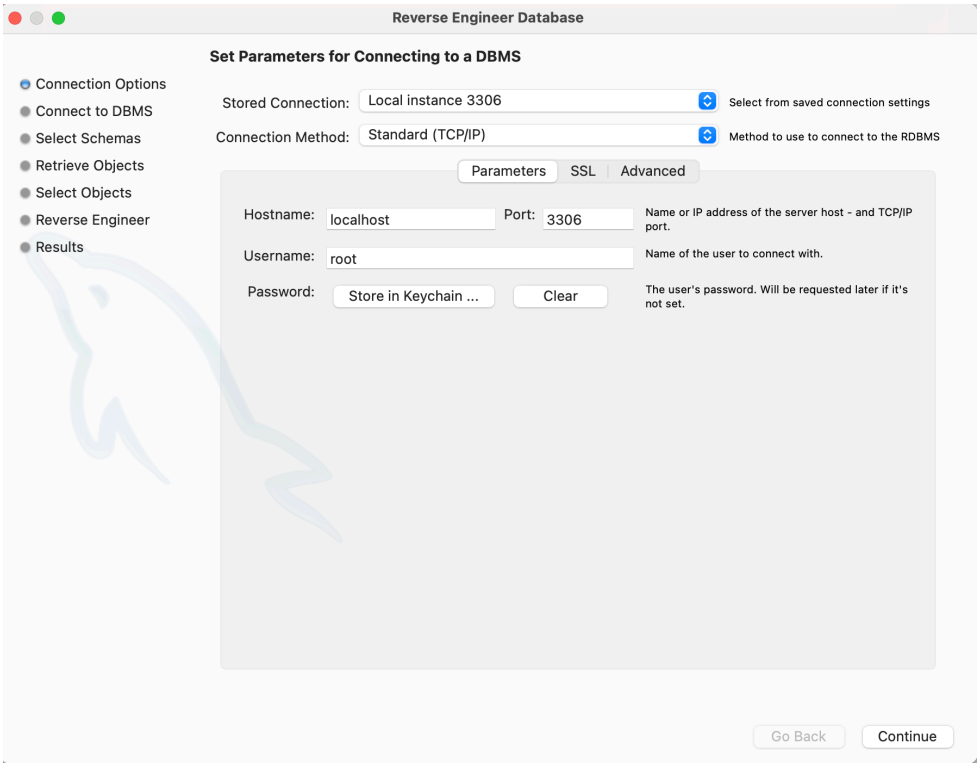
Entity-Relationship Diagram (ERD)

To gain a structured and comprehensive view of the OK_BRFFS_Survey database, I utilized the Reverse Engineer feature in MySQL Workbench to generate an Entity-Relationship Diagram (ERD). This ERD visually represents the relationships between tables, ensuring data integrity, reducing redundancy, and optimizing query performance. By reverse engineering the schema, I was able to obtain a clear representation of how the fact and dimension tables are linked within the database, helping in efficient data management.

The process began by navigating to the Database menu in MySQL Workbench and selecting Reverse Engineer.

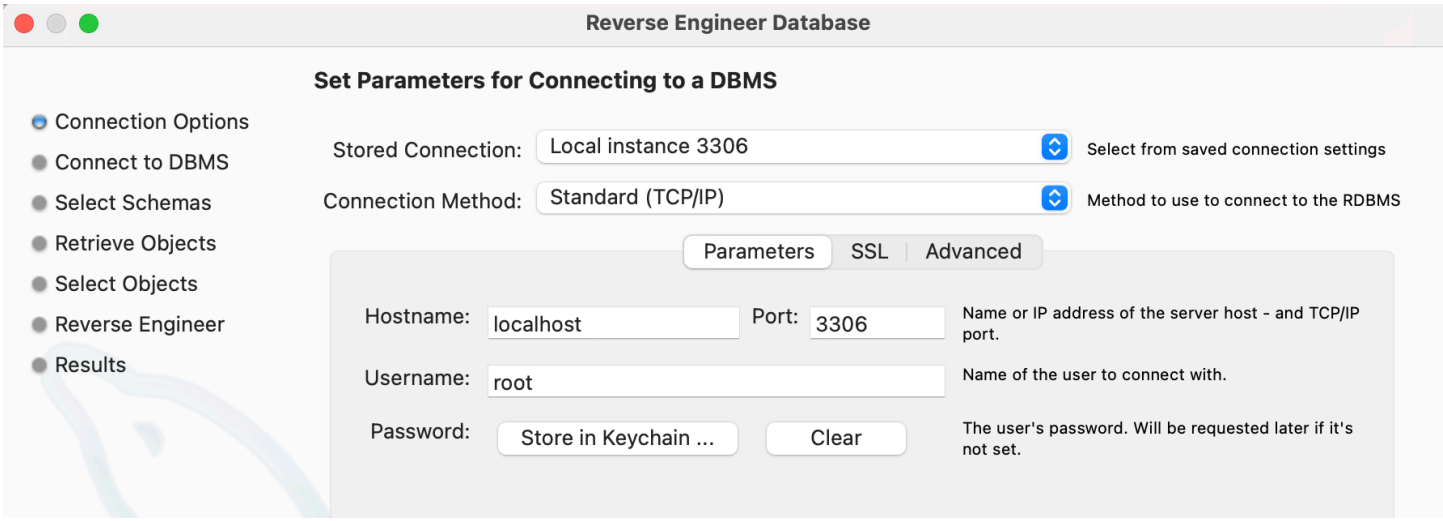


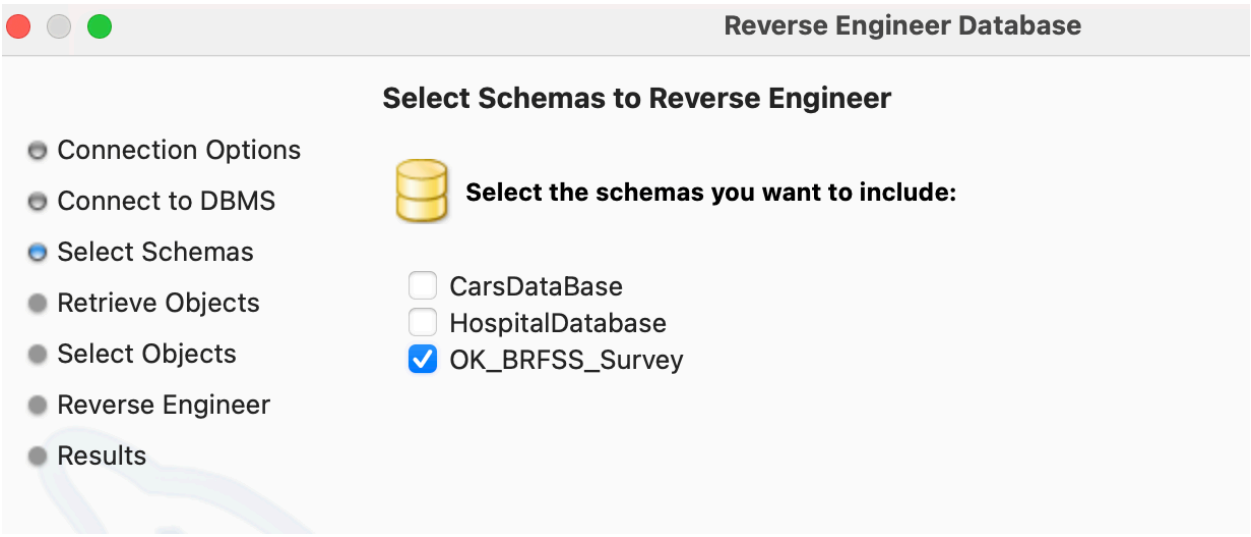
This feature allows MySQL Workbench to automatically extract the schema's structure, displaying tables, columns, and their relationships. Once I selected Reverse Engineer, I connected to the local MySQL server, using localhost as the hostname and root as the username.

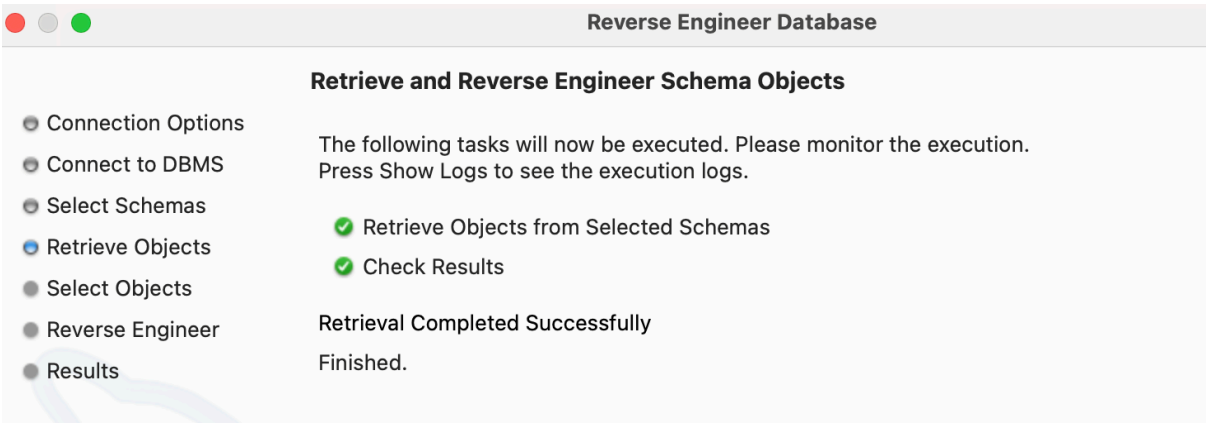


This connection was successfully established, allowing access to all available databases.

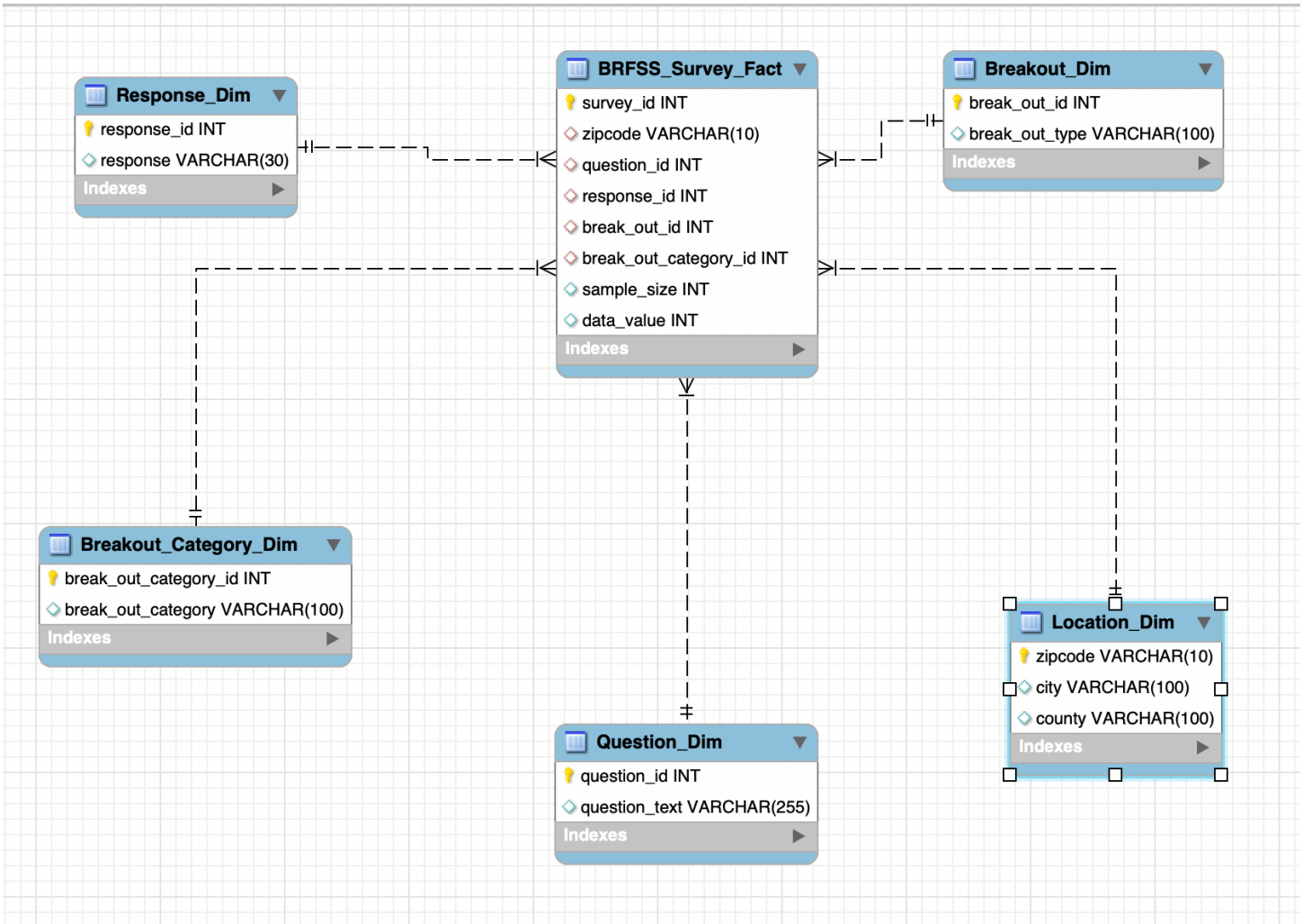
After successfully connecting, I selected the OK_BRFFS_Survey schema, which contains the relevant tables needed for analysis. The system then retrieved all database objects from this schema , ensuring that every table, index, and foreign key relationship was included. Following this, I selected all relevant tables to be reverse-engineered, ensuring that MySQL Workbench generated an accurate ERD representation . Once the retrieval process was completed, MySQL Workbench automatically generated the ERD, displaying the relationships between the tables in a structured format .







The final Entity-Relationship Diagram follows a star schema design, which is ideal for analytical processing. In this schema, the BRFSS_Survey_Fact table serves as the central fact table, which holds the core survey data, while multiple dimension tables surround it, providing contextual information. The fact table is linked to each dimension table through foreign keys, creating a one-to-many relationship between the fact and dimension tables.



Questions:

Question 1: Identifying Adolescent Age Groups at Highest Risk for Alcohol Abuse

To determine which adolescent age groups are most at risk for alcohol abuse, I designed an SQL query that focuses on the "Age Group" breakout category. Adolescents are generally defined as individuals between 10 and 19 years old, but for this analysis, I extended the range to 18-24 years to account for young adults who exhibit similar behavioral patterns. The query calculates the alcohol abuse percentage for each age group by dividing the total number of individuals who reported alcohol abuse (data_value) by the total number of participants (sample_size) and multiplying by 100. To prevent errors from division by zero, I used a CASE statement. The dataset was filtered to include only the 'Age Group' breakout category, and results were grouped and ordered by alcohol abuse ratio in descending order.

```sql
-- Question 1

-- Query to identify adolescent age groups at the highest risk for alcohol abuse.
-- This calculates the alcohol abuse percentage for each age group by dividing
-- the total number of positive responses by the total number of participants.

SELECT
    boc.break_out_category AS demographic_group, -- General category like "Age Group"
    bo.break_out_type AS age_group, -- Specific age range like "18-24"
    SUM(bsf.data_value) AS total_people_responded, -- Total people who reported alcohol abuse
    SUM(bsf.sample_size) AS total_participants, -- Total number of respondents
    CASE
        WHEN SUM(IFNULL(bsf.sample_size, 0)) = 0 THEN 0 -- Avoid division by zero
        ELSE ((SUM(IFNULL(bsf.data_value, 0)) / SUM(IFNULL(bsf.sample_size, 0))) * 100)
    END AS adolescent_alcohol_abuse_ratio -- Percentage of alcohol abuse cases
FROM
    BRFSS_Survey_Fact bsf
    JOIN Breakout_Dim bo ON bsf.break_out_id = bo.break_out_id
    JOIN Breakout_Category_Dim boc ON bsf.break_out_category_id = boc.break_out_category_id
WHERE
    boc.break_out_category = 'Age Group' -- Ensuring we filter correctly
GROUP BY
    boc.break_out_category, bo.break_out_type -- Group by category and specific age range
ORDER BY
    adolescent_alcohol_abuse_ratio DESC; -- Display from highest to lowest
```

The output shows that the 25-34 age group has the highest alcohol abuse ratio at 58.12%, followed by the 18-24 age group at 56.02%. While the 25-34 group is not classified under adolescence, the 18-24 age group, which includes individuals aged 18 and 19, is the highest-risk category within adolescents. The 35-44 age group follows with 49.41%, while older groups, 55-64 and 65+, show significantly lower alcohol abuse rates.

	demographic_gro...	age_group	total_people_respond...	total_participa...	adolescent_alcohol_abuse_r...
	Age Group	25-34	2481	4269	58.1167
	Age Group	18-24	2446	4366	56.0238
	Age Group	35-44	1978	4003	49.4129
	Age Group	55-64	2009	4308	46.6342
	Age Group	45-54	2083	4482	46.4748
	Age Group	65+	1763	4198	41.9962

These findings highlight the need for targeted intervention programs for young adults, particularly those aged 18-24. Universities and colleges should implement alcohol awareness programs to educate students on responsible drinking. Stricter ID verification policies and public awareness campaigns could help reduce alcohol access among high-risk individuals. Expanding mental health and counseling services for adolescents struggling with alcohol-related issues is also crucial.
By leveraging data-driven insights, policymakers and educational institutions can develop effective intervention strategies to address adolescent alcohol abuse, ensuring resources are allocated efficiently and contributing to a healthier, more informed society.

Question 2: Identifying the Areas of Oklahoma with the Highest and Lowest Percentage of Respondents for Adolescent Alcohol Abuse by City

To analyze adolescent alcohol abuse at a more granular level, I designed a query to identify the cities in Oklahoma with the highest and lowest alcohol abuse percentages among respondents aged 18-24. This analysis is crucial for understanding geographic trends in adolescent alcohol abuse, which can help target interventions in high-risk areas.

The query aggregates the total number of people who reported alcohol abuse (data_value) and the total number of survey participants (sample_size) for each city. The percentage of adolescent alcohol abuse is then calculated by dividing the number of positive responses by the total number of respondents and multiplying by 100. To ensure accuracy, I used the IFNULL function to prevent division by zero errors. The results are grouped by city and ordered in descending order to highlight the cities with the highest alcohol abuse percentages first.

To extract this information, I joined the BRFSS_Survey_Fact table with the Location_Dim table using zipcode to get city-wise data. Additionally, I joined the Breakout_Dim and Breakout_Category_Dim tables to filter only for the Age Group category and specifically for respondents aged 18-24, as this is the primary adolescent age range relevant to the study

```sql
-- Question 2  :Find the areas of Oklahoma with the highest and lowest percent
-- of respondents for adolescent alcohol abuse by city

-- Query to identify cities in Oklahoma with the highest and lowest adolescent alcohol abuse percentages.
-- This calculates the alcohol abuse ratio per city by dividing the total number of positive responses
-- by the total number of participants in that city.

SELECT
    ld.city, -- City in Oklahoma
    SUM(bsf.data_value) AS total_people_responded, -- Total people who reported alcohol abuse
    SUM(bsf.sample_size) AS total_participants, -- Total number of respondents
    CASE
        WHEN SUM(IFNULL(bsf.sample_size, 0)) = 0 THEN 0 -- Avoid division by zero
        ELSE ((SUM(IFNULL(bsf.data_value, 0)) / SUM(IFNULL(bsf.sample_size, 0))) * 100)
    END AS adolescent_alcohol_abuse_ratio -- Percentage of alcohol abuse cases
FROM
    BRFSS_Survey_Fact bsf
    JOIN Location_Dim ld ON bsf.zipcode = ld.zipcode -- Joining with location dimension table
    JOIN Breakout_Dim bo ON bsf.break_out_id = bo.break_out_id
    JOIN Breakout_Category_Dim boc ON bsf.break_out_category_id = boc.break_out_category_id
WHERE
    boc.break_out_category = 'Age Group' -- Ensuring we filter only for age group data
    AND bo.break_out_type = '18-24' -- Filtering for respondents aged 18-24
GROUP BY
    ld.city -- Grouping by city to get city-wise percentage
ORDER BY
    adolescent_alcohol_abuse_ratio DESC; -- Display from highest to lowest
```

The output indicates that Oklahoma City has the highest adolescent alcohol abuse percentage at **56.02%**, with 2,446 people reporting alcohol abuse out of 4,366 total respondents. This high percentage suggests that Oklahoma City could be an area of concern for underage drinking and adolescent substance abuse. Understanding these trends allows for better-targeted public health initiatives, such as awareness campaigns, intervention programs, and support systems for at-risk youth in the region.

city	total_people_respond...	total_participa...	adolescent_alcohol_abuse_r...
OKLAHOMA CITY	2446	4366	56.0238

The insights from this analysis can help policymakers, healthcare professionals, and community organizations focus their resources effectively to address adolescent alcohol abuse in high-risk cities. The ability to pinpoint specific locations enables more localized and effective intervention strategies, ultimately contributing to improved public health outcomes.

Question 3: Find the areas of Oklahoma with the highest and lowest percent of respondents for adolescent alcohol abuse by county.

To determine which counties in Oklahoma have the highest and lowest percentages of adolescent alcohol abuse, I designed a query that calculates the alcohol abuse ratio per county. This was done by dividing the total number of positive responses (data_value) by the total number of survey participants (sample_size) for each county. The percentage helps in identifying regional trends and potential areas that require targeted intervention.

The query begins by selecting the county from the Location_Dim table, ensuring that the results are grouped at the county level. The SUM function is used to aggregate the total number of individuals who reported alcohol abuse and the total number of participants within each county. To prevent division by zero errors, a CASE statement was implemented, ensuring that if the total sample size is zero, the percentage calculation does not result in an undefined value.

To ensure that only relevant data is considered, the query filters responses specifically for the "Age Group" category from the Breakout_Category_Dim table and further narrows it down to the "18-24" age group from the Breakout_Dim table. The JOIN operations effectively link the fact table (BRFSS_Survey_Fact) to the relevant dimension tables, allowing for a structured analysis of the data.

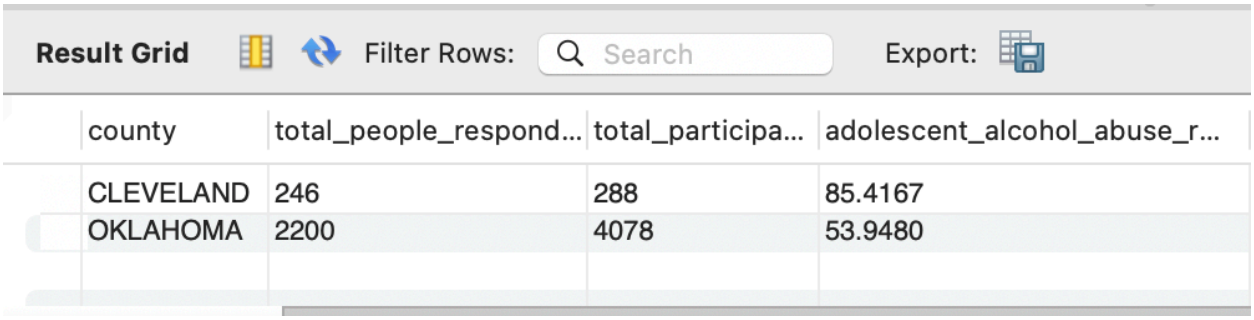
```sql
-- Question 3: Find the areas of Oklahoma with the highest and lowest percent
-- of respondents for adolescent alcohol abuse by county

-- Query to identify counties in Oklahoma with the highest and lowest adolescent alcohol abuse percentages.
-- This calculates the alcohol abuse ratio per county by dividing the total number of positive responses
-- by the total number of participants in that county.

SELECT
    ld.county, -- County in Oklahoma
    SUM(bsf.data_value) AS total_people_responded, -- Total people who reported alcohol abuse
    SUM(bsf.sample_size) AS total_participants, -- Total number of respondents
    CASE
        WHEN SUM(IFNULL(bsf.sample_size, 0)) = 0 THEN 0 -- Avoid division by zero
        ELSE ((SUM(IFNULL(bsf.data_value, 0)) / SUM(IFNULL(bsf.sample_size, 0))) * 100)
    END AS adolescent_alcohol_abuse_ratio -- Percentage of alcohol abuse cases
FROM
    BRFSS_Survey_Fact bsf
    JOIN Location_Dim ld ON bsf.zipcode = ld.zipcode -- Joining with location dimension table
    JOIN Breakout_Dim bo ON bsf.break_out_id = bo.break_out_id
    JOIN Breakout_Category_Dim boc ON bsf.break_out_category_id = boc.break_out_category_id
WHERE
    boc.break_out_category = 'Age Group' -- Ensuring we filter only for age group data
    AND bo.break_out_type = '18-24' -- Filtering for respondents aged 18-24
GROUP BY
    ld.county -- Grouping by county to get county-wise percentage
ORDER BY
    adolescent_alcohol_abuse_ratio DESC; -- Display from highest to lowest

```

After executing the query, the results indicate that Cleveland County has the highest adolescent alcohol abuse ratio at 85.41%, while Oklahoma County has a significantly lower rate of 53.95%. The stark difference between the counties highlights potential socio-economic, cultural, or environmental factors that might contribute to varying levels of adolescent alcohol abuse. These insights can be valuable for policymakers and public health officials in formulating targeted prevention programs or allocating resources more effectively.



county	total_people_respond...	total_participa...	adolescent_alcohol_abuse_r...
CLEVELAND	246	288	85.4167
OKLAHOMA	2200	4078	53.9480

The final results were sorted in descending order of alcohol abuse percentages, making it easy to identify the areas of concern. This approach provides a clear understanding of the regional variations in adolescent alcohol consumption, enabling further research into underlying causes and possible intervention strategies.

Conclusion

This project successfully achieved its goal of designing a well-structured data warehouse to analyze adolescent alcohol consumption trends in Oklahoma. By implementing a star schema model, I was able to optimize data storage, minimize redundancy, and improve query performance. The dimensional model provided a structured approach to extracting meaningful insights, allowing for efficient identification of high-risk age groups and geographic areas. Through structured SQL queries, I was able to answer key business questions regarding alcohol abuse rates by city, county, and age group. The results highlight the importance of using a properly normalized database for large-scale public health analysis.

Using MySQL for database management proved to be highly efficient for handling structured data, while MySQL Workbench provided a user-friendly interface for schema design and visualization. The ERD made it easier to understand the relationships between the fact and dimension tables, ensuring that the database adhered to best practices. However, one limitation was the reliance on SQL queries alone, which may not be the most efficient approach for advanced statistical analysis or predictive modeling. In future iterations, integrating Python with Pandas or R for statistical computations could enhance the depth of analysis by allowing for more

complex trend predictions and data visualizations. Additionally, improving the preprocessing step to include automated data validation could further enhance data integrity.

If I were to refine this project, I would focus on expanding the dataset to include more variables, such as socioeconomic status and external factors influencing alcohol consumption. Incorporating time-series analysis could also provide deeper insights into trends over multiple years, helping public health officials make more informed decisions. Lastly, optimizing queries for even faster execution, possibly through indexing strategies, would further enhance performance when working with large datasets. Overall, this project demonstrated the power of dimensional modeling for structured data analysis while highlighting potential areas for future improvement and expansion.