University of Florida

Computer and Information Science and Engineering

[COP5725] - Database Management Systems

Instructor: Dr. Markus Schneider

Crime Data Analysis for Safer Communities

Database Schema Construction

Project Group : 18

Group Members

Name	UFID
Venkata Suresh Yarava	61752296
Sri Charan Pabbathi	80984724
Chenna Kesava Varaprasad Korlapati	48368778
Sreenivasa Raju Konduru	83999388

Contents

1 Transformation of ER diagram

1.1 Description and Output of Complex Trend Queries

Based on feedback for *E-R Diagram and User Interface Design* documentation, following is description of the final complex trend queries, and description of output for complex trend queries.

- **Evolving Crime Trends :**
  - The present crime trend involves analyzing and understanding the current state of criminal activity.
  - It encompasses the types of crimes being committed, their frequency, and any patterns or changes that are evident in the most recent data.
  - A stacked bar chart is used to illustrate the trend across multiple years. In this chart, each stack represents a distinct type of crime, with years mapped on the horizontal (x) axis and crime rates on the vertical (y) axis.
- **Geographical Analysis of Safety and Security Trend :**
  - The present crime trend analysis involves examining recent criminal incidents in a specified geographic area.
  - This analysis provides insights into safety and security dynamics, including crime rates, types, and patterns over the years. The customized approach empowers stakeholders to make data-driven decisions for enhancing public safety.
  - A heat map is employed to depict the yearly trends, consisting of multiple boxes, each labeled with a specific year, and colored to indicate the severity or frequency of crimes occurring during that period.
- **Day-wise Crime Analysis Trend :**
  - The crime trend analysis involves studying the historical patterns of criminal activity over an extended period, specifically on different days of the week.
  - This detailed analysis helps identify temporal and situational patterns in crime rates, aiding law enforcement and policymakers in optimizing resources and strategies.

  - A multi-line chart is utilized to illustrate trends across the years, featuring seven lines, each representing a day of the week, with time periods marked on the x-axis and crime rates on the y-axis.

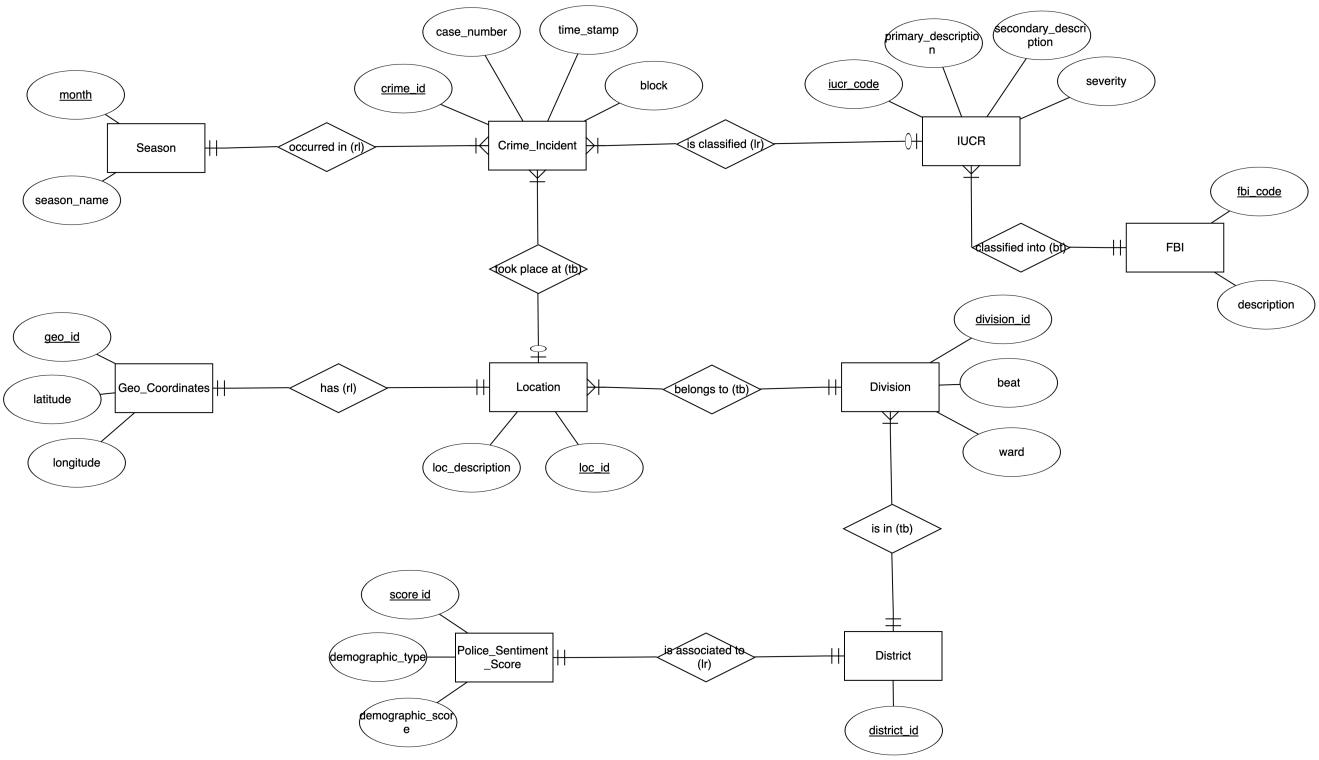
- **Season-Based Crime Analysis Trend**
  - Crime trend analysis is the examination of how criminal activity has evolved over an extended period. In this context, it entails a comprehensive study of crime rates across various types of locations over multiple years during different seasons.
  - This analysis helps identify spatial and temporal patterns in crime, enabling users to make informed decisions regarding safety and security.
  - A Choropleth Map is employed to depict trends across years in various locations. The color intensity signifies the frequency of occurrences, and it includes a filter for selecting specific season.

- **Dynamic Crime Analysis Trend :**
  - This analysis of crime trends involves a thorough investigation into how particular types of crimes have changed and developed over an extended period, with a specific focus on various time intervals.
  - This analysis allows for a customized, in-depth exploration of crime rates in user-specified locations and crime types over time.
  - A line chart is used to visualize the trend over the years.With time periods marked on the x-axis and crime rates on the y-axis.

- **Police Sentiment Analysis Trend :**
  - This crime trend analysis involves the examination of historical patterns and changes in criminal activity over a period of time.
  - It can provide valuable insights into the dynamics of crime, helping law enforcement, policymakers, and communities make informed decisions about safety and security.
  - A multi-line chart is employed to display long-term trends. It features time periods on the X-axis and maps police sentiment scores on the Y-axis, with each line representing the sentiment score for a specific district throughout the duration.

1.2 ER Diagram

Here's the updated ER Diagram.



Changes made in ER diagram from previous deliverable:

- Updated Police_Sentiment_Score entity's attributes.
- Renamed few entity names to maintain standard naming convention across all entities.
- Renamed few attribute names to maintain standard naming convention across all attributes.
- Renamed few relationship names to maintain standard naming convention across all relationships.

1.3 Transformation of ER diagram into Collection of Relation Schemas

Following is the collection of relation schemas obtained from above ER diagram along with explanation of transformation process.

- Crime_Incident(crime_id: integer, case_number: integer, time_stamp: date, block: string, iucr_code: string, loc_id: integer)

  Explanation: A separate relation schema is generated for the strong entity "Crime_Incident" which includes all its basic attributes along with their respective domains(data types) - crime_id: (integer), case_number: (integer), time_stamp: (date), block: (string). The names of the attributes in the schema mirror those of the entity, and the entity's key crime_id serves as the primary key in the schema, indicated by underlining. In addition, for binary n:1 relationship 'is classified' and n:1 relationship 'occured in', primary key of Location entity *loc_id* and primary key of IUCR entity *iucr_code* respectively are added as foreign keys to the relation.

- IUCR(iucr_code: string, primary_description: string, secondary_description: string, severity: string, fbi_code: string)

  Explanation: A separate relation schema is generated for the strong entity "IUCR" which includes all its basic attributes along with their respective domains(data types) - iucr_code: (string), primary_description: (string), secondary_description: (string), severity: (string). The names of the attributes in the schema mirror those of the entity, and the entity's key iucr_code serves as the primary key in the schema, indicated by underlining. In addition, for binary n:1 relationship 'classified into', primary key of FBI entity *fbi_code* is added as foreign key to the relation.

- FBI(fbi_code: string, description: string)

  Explanation: A separate relation schema is generated for the strong entity "FBI" which includes all its basic attributes along with their respective domains(data types) - fbi_code: (string), description: (string). The names of the attributes in the schema mirror those of the entity, and the entity's key fbi_code serves as the primary key in the schema, indicated by underlining.

- Season(month: integer, season_name: string)

  Explanation: A separate relation schema is generated for the strong entity "Season" which includes all its basic attributes along with their respective domains(data types) - month: (integer), season_name: (string). The names of the attributes in the schema mirror those of the entity, and the entity's key month serves as the primary key in the schema, indicated by underlining.

- Location(loc_id: integer, loc_description: string, division_id: integer, geo_id: integer)

  Explanation: A separate relation schema is generated for the strong entity "Location" which includes all its basic attributes along with their respective domains(data types) - loc_id: (integer), loc_description: (string). The names of the attributes in the schema mirror those of the entity, and the entity's key loc_id serves as the primary key in the schema, indicated by underlining. In addition, for binary n:1 relationship 'belongs to' and 1:1 relationship 'has', primary key of Division entity *division_id* and primary key of Geo_Coordinates entity *geo_id* respectively are added as foreign keys to the relation.

- Geo_Coordinates(geo_id: integer, latitude: string, longtitude: string)

  Explanation: A separate relation schema is generated for the strong entity "Geo_Coordinates" which includes all its basic attributes along with their respective domains(data types) - geo_id: (integer), latitude: (string), longitude: (string). The names of the attributes in the schema mirror those of the entity, and the entity's key geo_id serves as the primary key in the schema, indicated by underlining.

- Division(division_id: integer, ward: string, beat: string, district: string)

  Explanation: A separate relation schema is generated for the strong entity "Division" which includes all its basic attributes along with their respective domains(data types) - division_id: (integer), ward: (string), beat: (string). The names of the attributes in the schema mirror those of the entity, and the entity's key division_id serves as the primary key in the schema, indicated by underlining. In addition, for binary n:1 relationship 'is in', primary key of District entity *district_id* is added as foreign key *district* to the relation.

- District(district_id: string)

  Explanation: A separate relation schema is generated for the strong entity "District" which includes single attribute along with its domain - district_id: string. The name of the attribute in the schema mirror that of the entity, and the entity's key district_id serves as the primary key in the schema, indicated by underlining.

- Police_Sentiment_Score(score_id: integer, district: string, demographic_type: string, demographic_score: string)

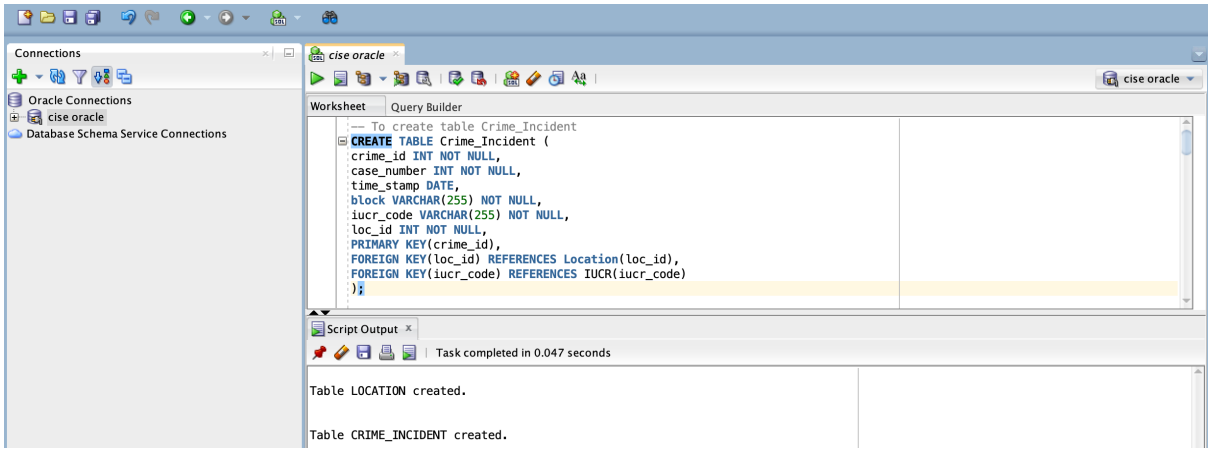
  Explanation: A separate relation schema is generated for the strong entity "Police_Sentiment_Score" which includes all its basic attributes along with their respective domains(data types) - score_id: (integer), demographic type: (string), demographic_score: (string). The names of the attributes in the schema mirror those of the entity, and the entity's key score_id serves as the primary key in the schema, indicated by underlining. In addition, for binary 1:1 relationship 'is associated to', primary key of District entity *district_id* is added as foreign key *district* to the relation.

2 Transformation of Relation Schemas

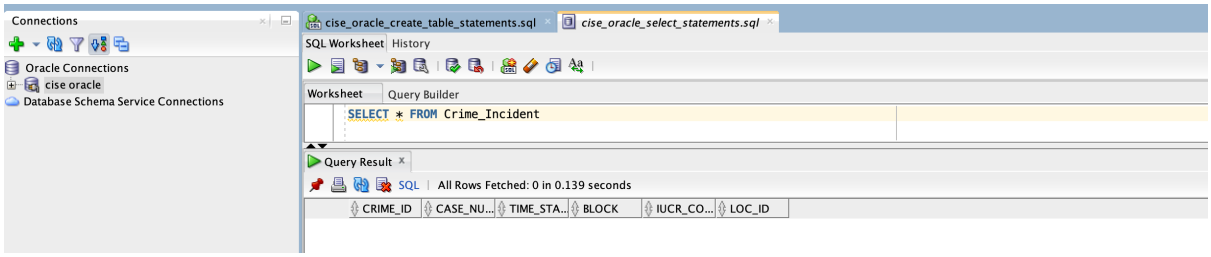
2.1 Transformation of Relation Schemas into SQL table schemas

SQL table schemas were created from above generated relation schemas using the create table command. Following are the screenshots of create table commands and created tables in Oracle SQL Developer.

Crime_Incident:

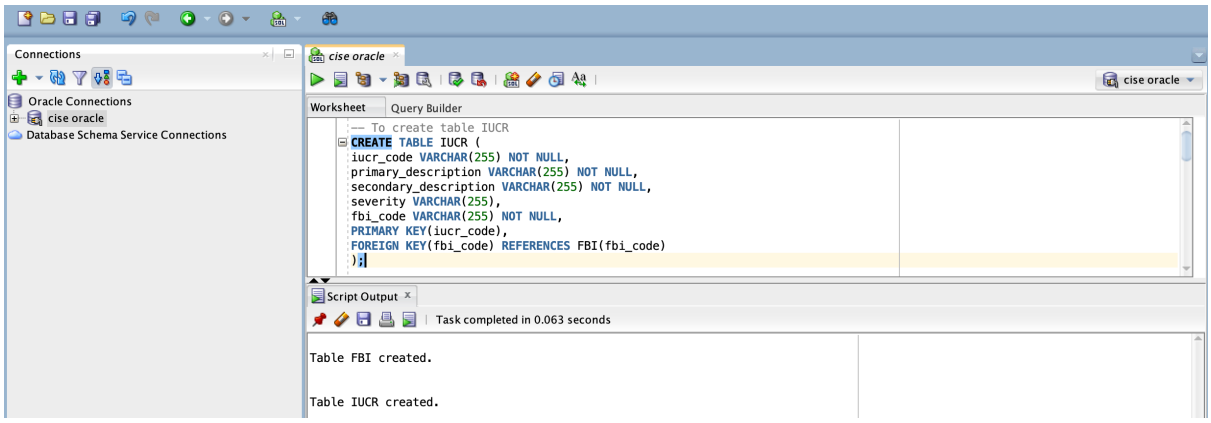


Create Table

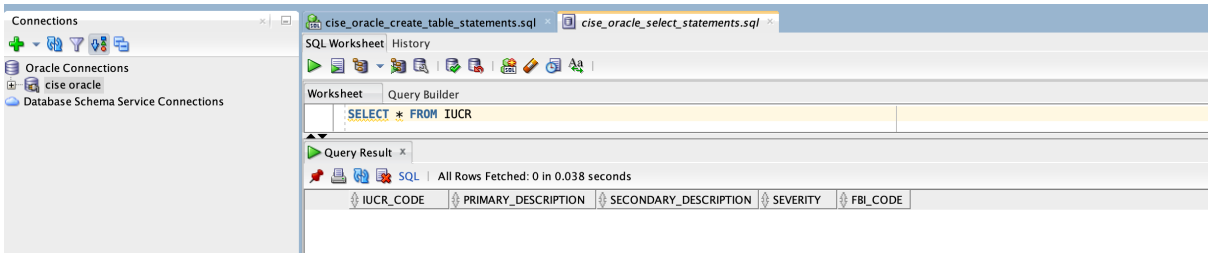


Empty Table

IUCR:

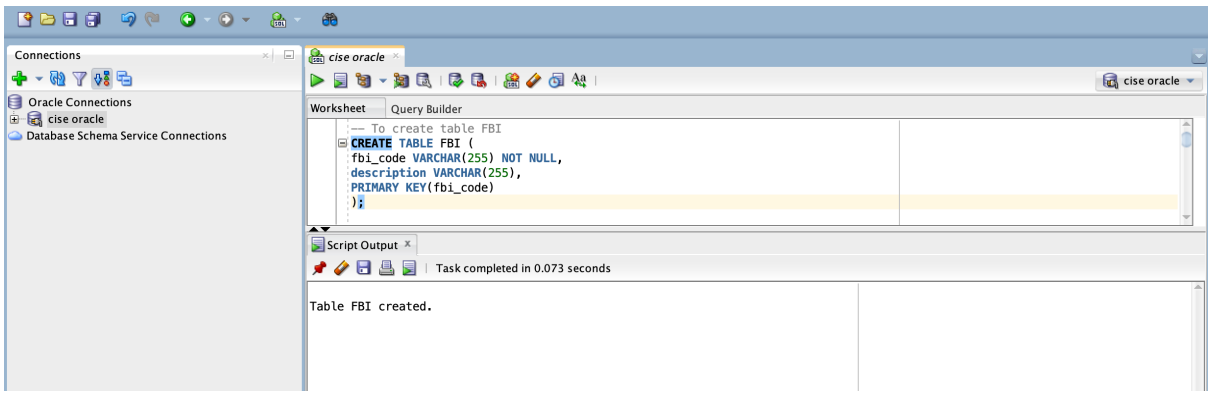


Create Table

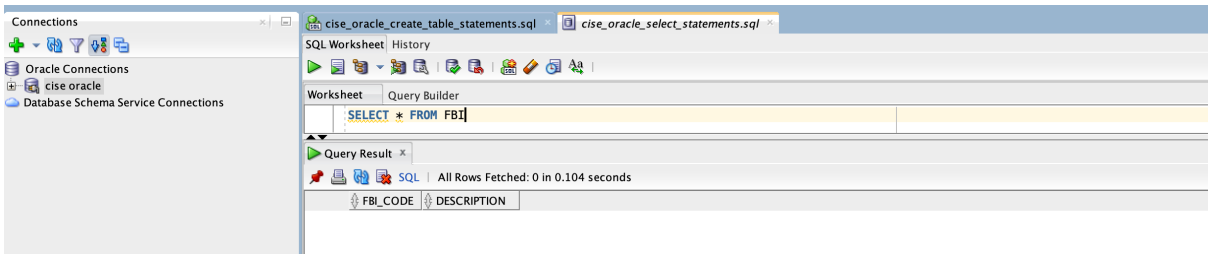


Empty Table

FBI:

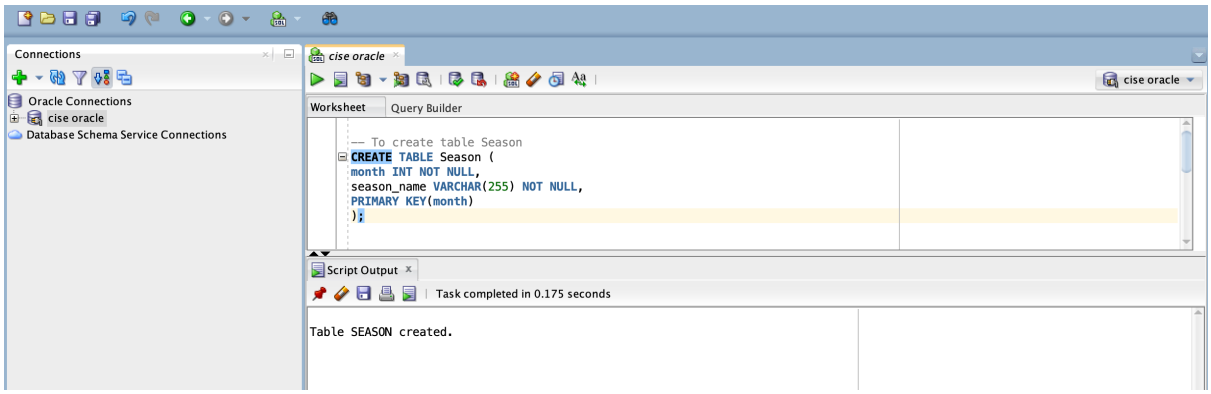


Create Table

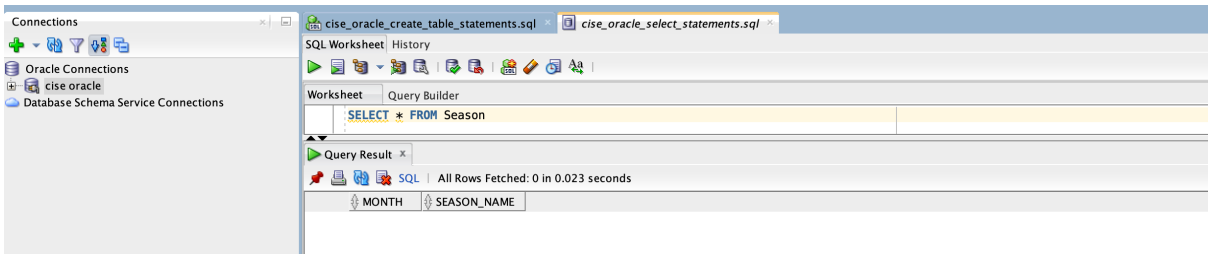


Empty Table

Season:

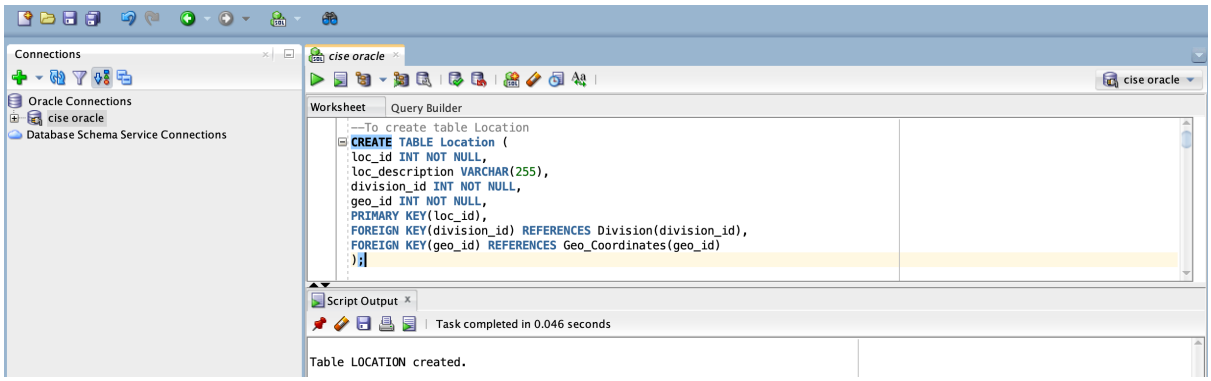


Create Table

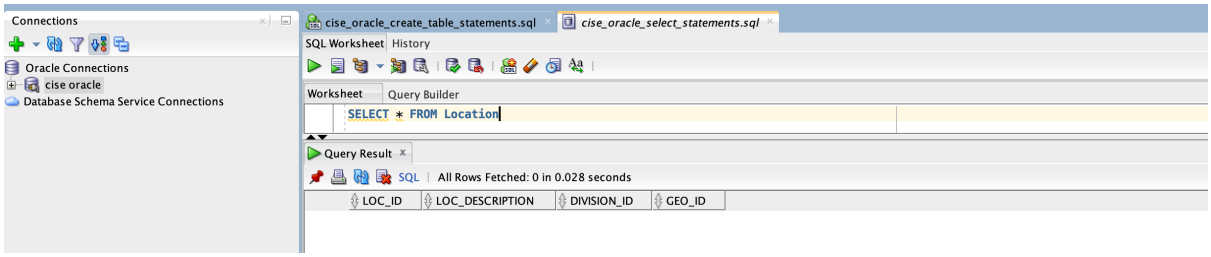


Empty Table

Location:

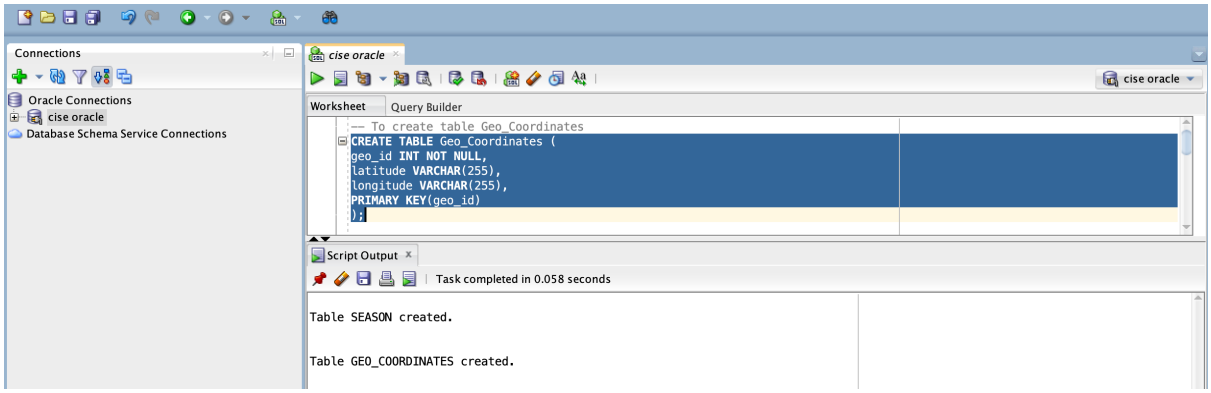


Create Table

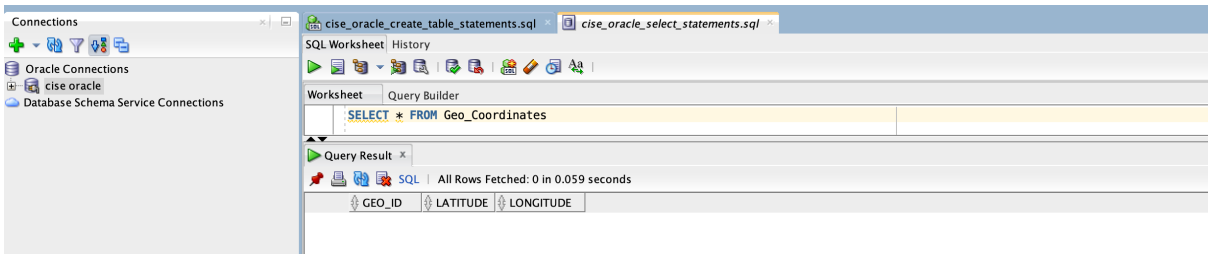


Empty Table

Geo_Coordinates:

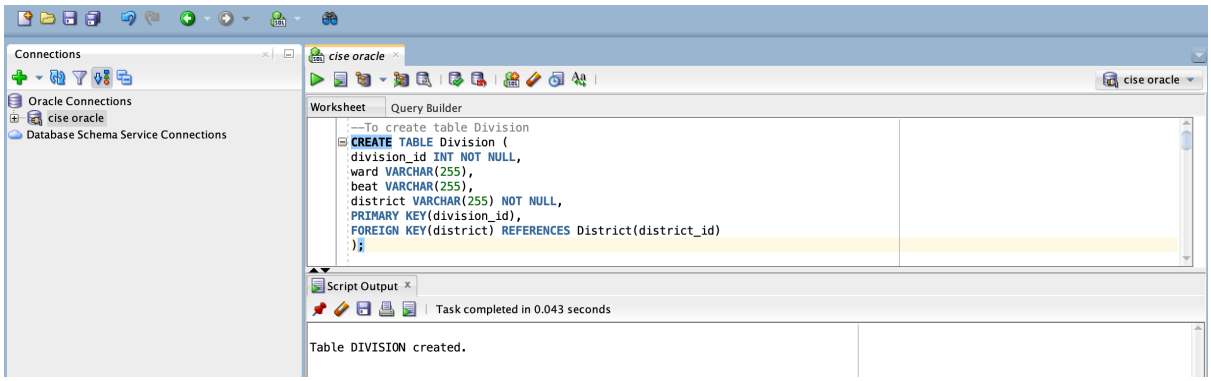


Create Table

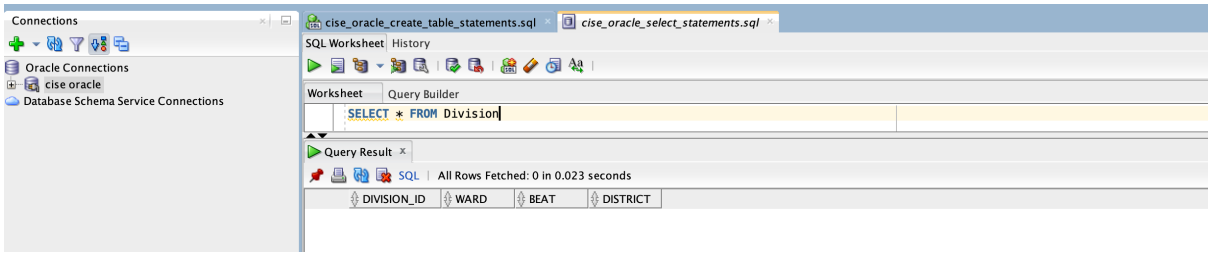


Empty Table

Division:

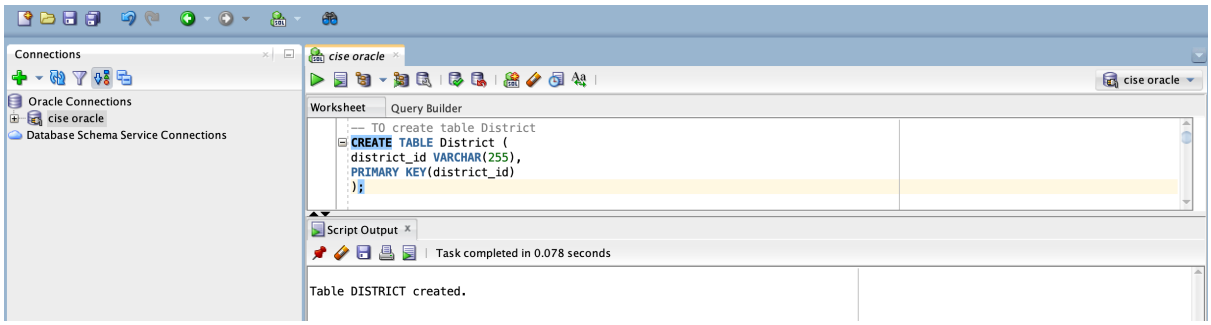


Create Table

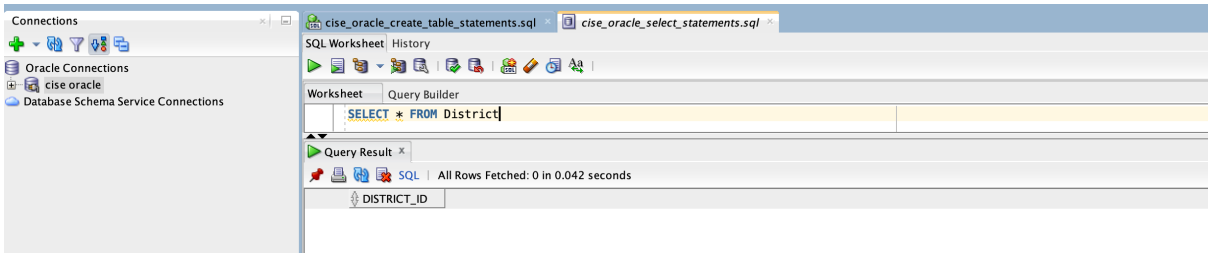


Empty Table

District:

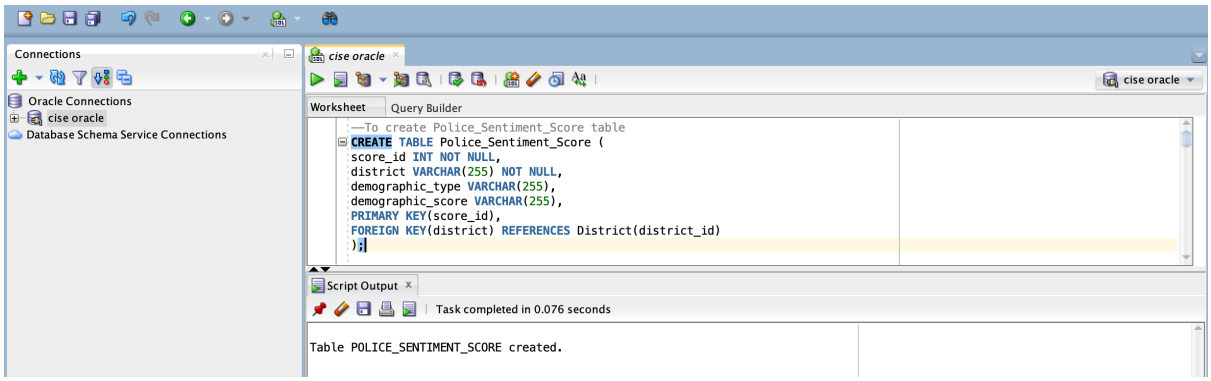


Create Table

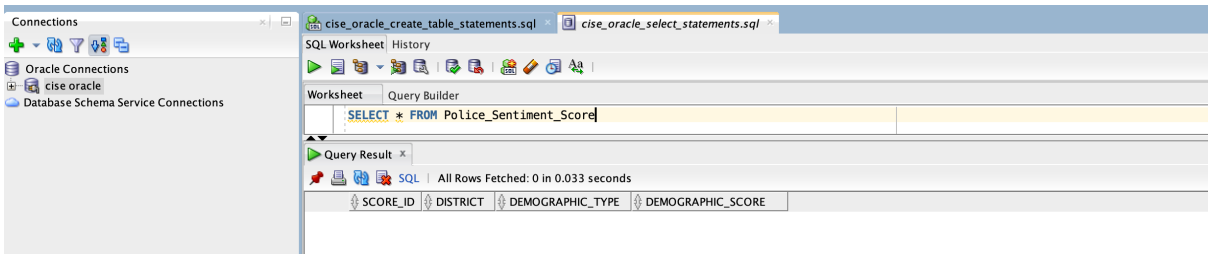


Empty Table

Police_Sentiment_Score:



Create Table



Empty Table

2.2 Explanation: Transformation of Relation Schemas into SQL table schemas

Following is the collection of sql schemas created from collection of relation schemas along with explanation of transformation process.

- **Crime_Incident table**

  Relation Schema: Crime_Incident(crime_id: integer, case_number: integer, time_stamp: date, block: string, iucr_code: string, loc_id: integer)

  SQL Schema:

```
CREATE TABLE Crime_Incident (
    crime_id INT NOT NULL,
    case_number INT NOT NULL,
    time_stamp DATE,
    block VARCHAR(255) NOT NULL,
    iucr_code VARCHAR(255) NOT NULL,
    loc_id INT NOT NULL,
    PRIMARY KEY(crime_id),
    FOREIGN KEY(loc_id) REFERENCES Location(loc_id),
    FOREIGN KEY(iucr_code) REFERENCES IUCR(iucr_code)
    );
```

  Explanation: A table titled 'Crime_Incident' is generated as part of conversion of the relation schema into its SQL schema. The following columns are present in the table:

  - crime_id (INT): Primary key of this table
  - case_number (INT): Represents case-specific identifier for each incident related case

- time_stamp (DATE): Represents date and time of crime incident
- block (VARCHAR(255)): Represents partially redacted address where incident occured
- iucr_code (VARCHAR(255)): Represents unique IUCR code associated with crime-incident. Added as foreign key from IUCR table. Can be used to fetch additional details about IUCR code of incident through a join with IUCR table
- loc_id (INT): Represents unique identifier for location of crime incident. Added as foreign key from Location table. Can be used to fetch additional details about location of incident through a join with Location table

- **IUCR table**

Relation Schema: IUCR(iucr_code: string, primary_description: string, secondary_description: string, severity: string, fbi_code: string)

SQL Schema:

```
CREATE TABLE IUCR (
    iucr_code VARCHAR(255) NOT NULL,
    primary_description VARCHAR(255) NOT NULL,
    secondary_description VARCHAR(255) NOT NULL,
    severity VARCHAR(255),
    fbi_code VARCHAR(255) NOT NULL,
    PRIMARY KEY(iucr_code),
    FOREIGN KEY(fbi_code) REFERENCES FBI(fbi_code)
    );
```

Explanation: A table titled 'IUCR' is generated as part of conversion of the relation schema into its SQL schema. The following columns are present in the table:

- iucr_code (VARCHAR(255)): Primary key of this table. Represents the unique IUCR (Illinois Uniform Crime Reporting) code associated with different crime types.
- primary_description (VARCHAR(255)) : Represents description of specific offense corresponding to the IUCR code
- secondary_description (VARCHAR(255)): Represents additional description of spe-

cific offense corresponding to the IUCR code

- severity (VARCHAR(255)): Represents level of seriousness associated with a particular offense corresponding to the IUCR code
- fbi_code (VARCHAR(255)): Represents unique FBI code associated with IUCR code. Added as foreign key from FBI table. Can be used to fetch additional details about FBI code of incident through a join with FBI table

- **FBI table**

  Relation Schema: FBI(fbi_code: string, description: string)

  SQL Schema:

```
CREATE TABLE FBI (
    fbi_code VARCHAR(255) NOT NULL,
    description VARCHAR(255),
    PRIMARY KEY(fbi_code)
    );
```

  Explanation: A table titled 'FBI' is generated as part of the conversion of the relation schema into its SQL schema. The following columns are present in the table:

  - fbi_code (VARCHAR(255)): Primary key of this table. Represents unique FBI code associated with different crime types
  - description (VARCHAR(255)): Represents description of specific offense corresponding to the FBI code

- **Season table**

  Relation Schema: Season(month: integer, season_name: string)

  SQL Schema:

```
CREATE TABLE Season (
    month INT NOT NULL,
    season_name VARCHAR(255) NOT NULL,
    PRIMARY KEY(month)
    );
```

Explanation: A table titled 'Season' is generated as part of conversion of the relation schema into its SQL schema. The following columns are present in the table:

- month (INT): Primary key of this table. Represent a unique month in a year
- season_name (VARCHAR(255)): Represents season name associated with a month.

- **Location table**

Relation Schema: Location(loc_id: integer, loc_description: string, division_id: integer, geo_id: integer)

SQL Schema:

```
CREATE TABLE Location (
    loc_id INT NOT NULL,
    loc_description VARCHAR(255),
    division_id INT NOT NULL,
    geo_id INT NOT NULL,
    PRIMARY KEY(loc_id),
    FOREIGN KEY(division_id) REFERENCES Division(division_id),
    FOREIGN KEY(geo_id) REFERENCES Geo_Coordinates(geo_id)
    );
```

Explanation: A table titled 'Location' is generated as part of conversion of the relation schema into its SQL schema. The following columns are present in the table:

- loc_id (INT): Primary key of this table. Represents a unique identifier for each tuple in this table
- loc_description (VARCHAR(255)): Represents description of specific location
- division_id (INT): Represents unique division_id associated with each location. Added as foreign key from Division table. Can be used to fetch additional details about division of a location through a join with Division table
- geo_id(INT): Represents unique geo_id associated with each location. Added as foreign key from Geo_Coordinates table. Can be used to fetch additional details about geo_coordinates of a location through a join with Geo_Coordinates table

- **Geo_Coordinates table**

  Relation Schema: Geo_Coordinates(geo_id: integer, latitude: string, longtitude: string)

  SQL Schema:

```
CREATE TABLE Geo_Coordinates (
    geo_id INT NOT NULL,
    latitude VARCHAR(255),
    longitude VARCHAR(255),
    PRIMARY KEY(geo_id)
    );
```

  Explanation: A table titled 'Geo_Coordinates' is generated as part of the conversion of the relation schema into its SQL schema. The following columns are present in the table:

  - geo_id (INT): Primary key of this table. Represents a unique identifier for each tuple in this table
  - latitude (VARCHAR(255)): Represents latitude coordinate associated with a location
  - longitude (VARCHAR(255)): Represents longitude coordinate associated with a location

- **District table**

  Relation Schema:District(district_id: string)

  SQL Schema:

```
CREATE TABLE DISTRICT(
district_id VARCHAR(255)
PRIMARY KEY(district_id)
);
```

  Explanation: A table titled 'District' is generated as part of conversion of the relation schema into its SQL schema. The following columns are present in the table:

  - district_id (VARCHAR(255)): Primary key of this table. Represents unique district name

- **Division table**

  Relation Schema: Division(division_id: integer, ward: string, beat: string, district: string)

  SQL Schema:

```
CREATE TABLE Division (
    division_id INT NOT NULL,
    ward VARCHAR(255),
    beat VARCHAR(255),
    district VARCHAR(255) NOT NULL,
    PRIMARY KEY(division_id),
    FOREIGN KEY(district) REFERENCES District(district_id)
    );
```

  Explanation: A table titled 'Division' is generated as part of conversion of the relation schema into its SQL schema. The following columns are present in the table:

  - division_id (INT): Primary key of this table. Represents a unique identifier for each division
  - ward (VARCHAR(255)): Represents wards associated with each division
  - beat (VARCHAR(255)): Represents beats under each division. A beat is the smallest police geographic area
  - district (VARCHAR(255)): Represents unique district associated with each division. Added as foreign key from District table

- **Police_Sentiment_Score table**

  Relation Schema: Police_Sentiment_Score(score_id: integer, district: string, demographic_type: string, demographic_score: string)

  SQL Schema:

```
CREATE TABLE Police_Sentiment_Score (
    score_id INT NOT NULL,
    district VARCHAR(255) NOT NULL,
    demographic_type VARCHAR(255),
```

```
    demographic_score VARCHAR(255),
    PRIMARY KEY(score_id),
    FOREIGN KEY(district) REFERENCES District(district_id)
);
```

Explanation: A table titled 'Police_Sentiment_Score ' is generated as part of conversion of the relation schema into its SQL schema. The following columns are present in the table:

- score_id (INT): Primary key of this table. Represents a unique identifier for each tuple in this table
- district (VARCHAR(255)): Represents unique district associated with each score. Added as foreign key from District table
- demographic_type (VARCHAR(255): Represents classification of individuals based on different demographic characteristics like Age, Gender, Race, Education, Income etc
- demographic_score (VARCHAR(255)): Represents trust score of associated demographic_type within each district. It indicates level of trust that people have in police