



School of Science, Engineering and Environment.

M.Sc. in Data Science

Assessment Title:

ADVANCED DATABASE ASSIGNMENT

Module Title: Advanced Database for Msc.

Semester: 2

Programme Code: G430 M0052

Name: Stella Awoyomi

Student ID: @00655282

Tutor: Prof. Mo Saraee, Dr Charith Silva

Table of Contents

Task1.....	4
Abstract.....	4
Introduction.....	4
Relational Schema.....	4
Design Rationale.....	8
Design Considerations.....	8
T-SQL Statement	10
Database Security	18
Database Backup and Restore Strategy	19
Data Science/Business Intelligence Techniques	19
Data Privacy, Ethical and legal issues	19
Conclusion.....	19
Task 2.....	20
Abstract.....	20
Introduction.....	20
Design Rationale.....	20
Design Considerations.....	20
T-SQL Statement	20
Report Design	26
Database Backup and Restore Strategy	29
Conclusion.....	30
Task 3.....	30

Abstract.....	30
Introduction.....	30
Design Rationale.....	31
Design Considerations.....	31
T-SQL Statement	31
Report Design	35
Database Backup and Restore Strategy	40
Data Science/Business Intelligence Techniques	40
Conclusion.....	40
References.....	41

TASK 1

The design and Implementation using T- sql statement to create various summarized reports that allows the measure of child poverty in Ethiopia and India.

1. Abstract

Dataset used consist of only data of two countries which are Ethiopia and India. These data was acquired from an ongoing research on the child well-being monitoring and reporting system carried out by Young Lives which is an international study of childhood poverty, following the lives of 12,000 children in the four countries (Ethiopia, India, Peru and Vietnam) over 15 years. The research is to understand Child poverty in these lower income countries. The purpose of the project is to improve understanding of the causes and consequences of childhood poverty and examine how policies affect children's well-being, in order to inform the development of future policy and to target child welfare interventions more effectively. The objective of this task is to create and design a database system that gives report to measure the child poverty in Ethiopia and India. Tables, views, stored procedure and other functions were created in SQL Server to help achieve this.

2. Introduction

The Young Lives study aims to track the lives of 12,000 children over 15 years, surveyed once every 3- 4 years. Round 1 of Young Lives surveyed two groups of children in each country, at 1 year old and 5 years old. Round 2 returned to the same children who were then aged 5 and 12 years old. Round 3 surveyed the same children again at aged 7-8 years and 14-15 years, Round 4 surveyed them at 12 and 19 years old, and Round 5 surveyed them at 15 and 22 years old. The objective of this task is to create and design a database system that gives report to measure the child poverty in Ethiopia and India.

3. Relational Schema

3.1 Identify the tables

The records of the tables was acquired from an ongoing research on the child well-being monitoring and reporting system carried out by Young Lives which is an international study of childhood poverty,

following the lives of 12,000 children. These report is concentrating on two countries (Ethiopia and India).

3.2 Creating the tables

The datasets for the two countries were downloaded as tab files from the UK Data Service website. These files are `ethiopia_constructed` and `india_constructed`. The database is named “Child_Poverty”. The two datasets are inserted into the `Child_Poverty` database. The inbuilt schema is “`dbo`”. The tables are created on the MS SQL Server with the schema name `dbo.ethiopia_constructed` and `dbo.india_constructed`.

Thirteen tables were created by means of the data dictionary from the two countries’ datasets. These tables were created with the ‘Schema’ using different column names to show the relationships between the tables created.

The Thirteen tables are:

1. `dbo.Birth_and_Immunisations`
2. `dbo.Child_Anthropometric`
3. `dbo.Child_Caregiver_Info`
4. `dbo.Child_Educations`
5. `dbo .Child_General_Feature`
6. `dbo.Child_Health_and_Wellbeing`
7. `dbo.Child_Identity_and_Location`
8. `dbo.Child_Parent`
9. `dbo.Child_Reproductive_Health`
10. `dbo.Child_Smokes_and_Drink`
11. `dbo.Child_Time`
12. `dbo.ChildInjury`
13. `dbo.ChildRead_and_Writes`

3.3 Creating relationship between tables

dbo.Child_Identity_and_Location' table was the primary table and the columns used in creating this table include childid, round, clustid, commid, typesite, region, yc, childloc, inround, panel and deceased. The childid and round columns are the primary keys. The childid and round columns are used as foreign keys to create the remaining tables while referencing the primary table and adding the necessary constraints. The dbo.Child_Identity_and_Location table comprises variables that define each child that was surveyed in all the rounds during the period. The relational schema for all the tables created and connection to the primary key are shown in Figure 3.1 below.

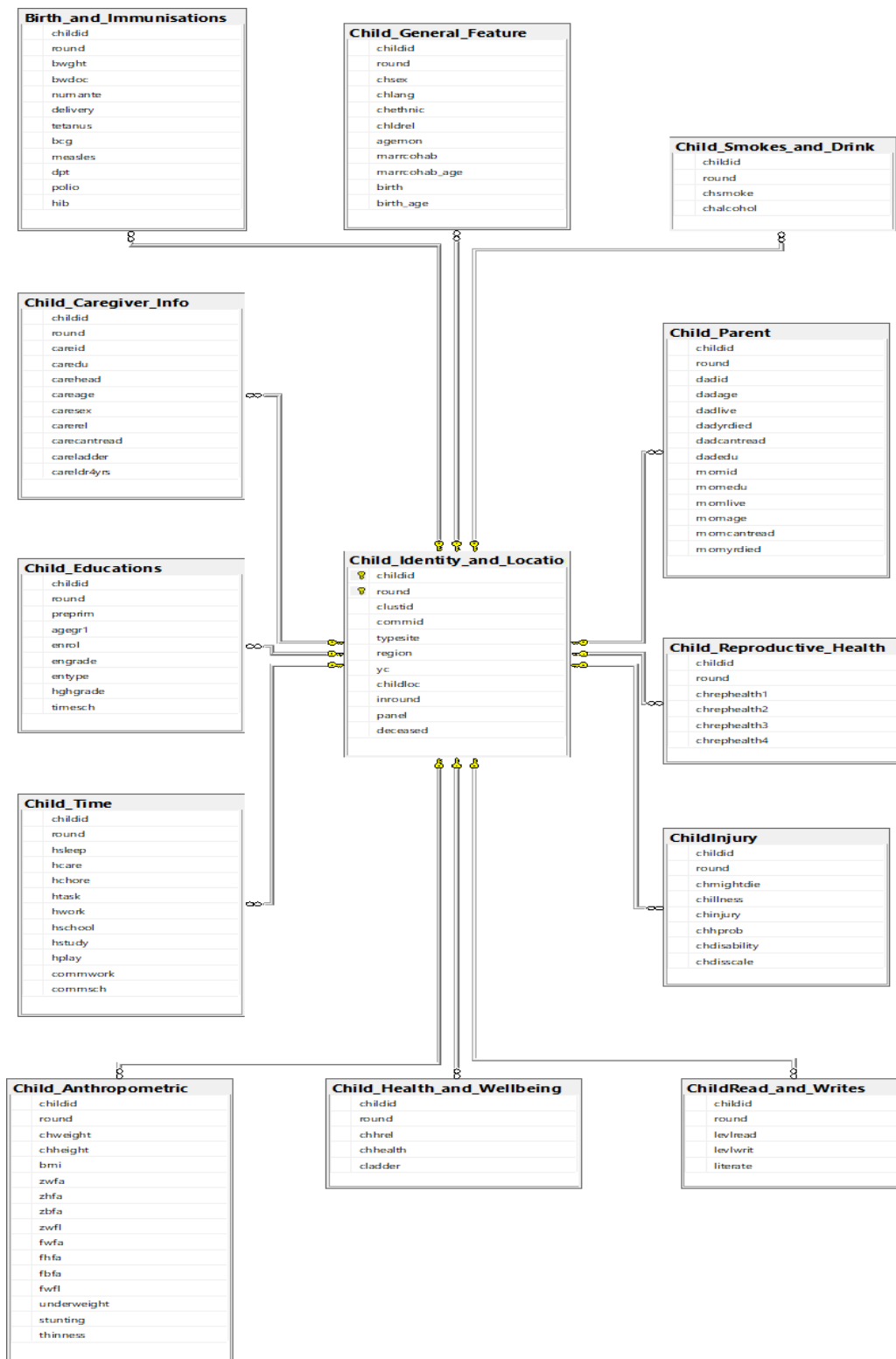


Figure 3.1 The relational schema for all the tables

4. Design Rationale

The right design is needed to accomplish one's objectives when working with a database. This task is concern with distributing the information into various tables to lessen redundant data; provide the SQL server with the facts it requires to join the information in the tables together and by means of the right data values and datatypes.

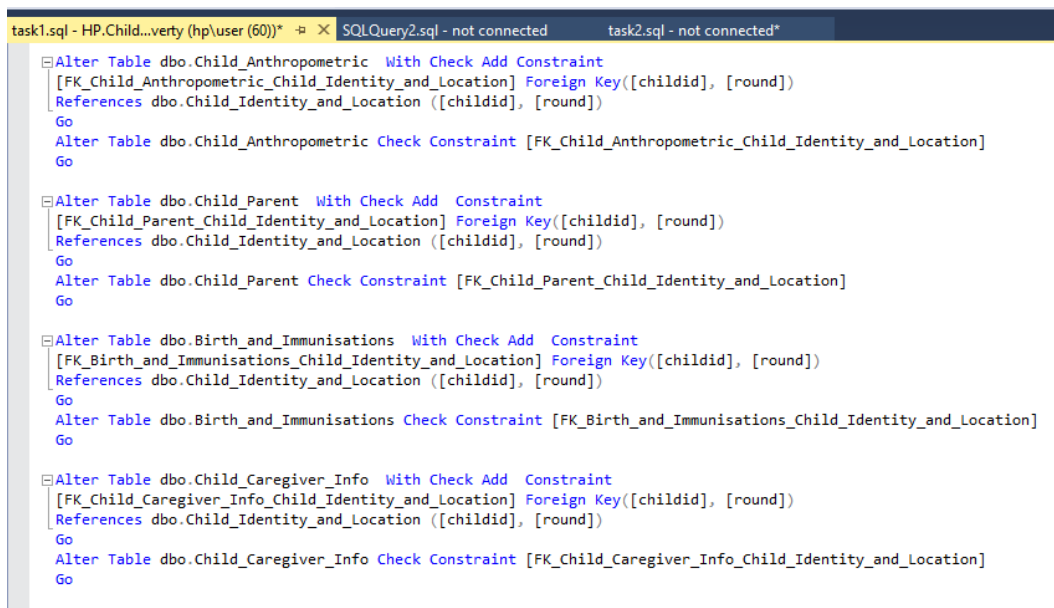
5. Design Considerations

5.1 Database Normalization

Data normalization rules are used to establish that the tables created are arranged appropriately.

5.2 Constraints

The childid and round were used as primary key on the 'Child_Identity_and_Location' table and set as foreign key 'with check add constraints' in generating other tables. This is shown in the Figure 5.2.1 below



```
task1.sql - HP.Child...verty (hp\user (60)) *  X  SQLQuery2.sql - not connected  task2.sql - not connected*

Alter Table dbo.Child_Anthropometric With Check Add Constraint
[FK_Child_Anthropometric_Child_Identity_and_Location] Foreign Key([childid], [round])
References dbo.Child_Identity_and_Location ([childid], [round])
Go
Alter Table dbo.Child_Anthropometric Check Constraint [FK_Child_Anthropometric_Child_Identity_and_Location]
Go

Alter Table dbo.Child_Parent With Check Add Constraint
[FK_Child_Parent_Child_Identity_and_Location] Foreign Key([childid], [round])
References dbo.Child_Identity_and_Location ([childid], [round])
Go
Alter Table dbo.Child_Parent Check Constraint [FK_Child_Parent_Child_Identity_and_Location]
Go

Alter Table dbo.Birth_and_Immunisations With Check Add Constraint
[FK_Birth_and_Immunisations_Child_Identity_and_Location] Foreign Key([childid], [round])
References dbo.Child_Identity_and_Location ([childid], [round])
Go
Alter Table dbo.Birth_and_Immunisations Check Constraint [FK_Birth_and_Immunisations_Child_Identity_and_Location]
Go

Alter Table dbo.Child_Caregiver_Info With Check Add Constraint
[FK_Child_Caregiver_Info_Child_Identity_and_Location] Foreign Key([childid], [round])
References dbo.Child_Identity_and_Location ([childid], [round])
Go
Alter Table dbo.Child_Caregiver_Info Check Constraint [FK_Child_Caregiver_Info_Child_Identity_and_Location]
Go
```

Figure 5.2.2 Child_Identity_and_Location set as foreign key with check add constraint

5.3 Data Validation

The datasets used for this task was acquired from the Young Lives project and the UK Data Service. This assures the value of foundation and correctness of the data used. The datasets were in a tab file format but were converted into an xlsx format before usage. The datatypes assigned to the variables of the dataset while importing into MS SQL conforms to the metadata dictionary.

5.4 Security

Enforcing security is very crucial in the design of a database design. Two roles were created using the select, insert and update access for the HODs in Education and Health unit. This is shown in figure 5.4.1 below

```
/**Enforcing security for HOD of departments**/  
/**Education HOD only**/  
Create role EducationHOD  
Grant select, insert, update on dbo.vGenEducationHistory to  
EducationHOD  
  
/**Health HOD only**/  
Create role HealthHOD  
Grant select, insert, update on [dbo].[vChild_Health_History] to  
HealthHOD
```

Figure 5.4.1 Enforcing security for Education and Health HODS

5.5 Comments

Comments are written for straightforwardness of the queries and what tasks are being performed at each step.

6. T-SQL Statements

6.1 Tables

For this task, thirteen tables were generated from the datasets using the data dictionaries. The first and most significant table is centered on the child's identifiers and location. This table is named `Child_Identity_and_Location` and assigned the primary table with primary keys set on the `childid` and `round` variables. Other child identifier variables used to create the table are the cluster ID, community ID, area of residence; whether rural or urban, region, cohort, child location and if the child is deceased during the period of the survey. These variables are represented by `clustid`, `commid`, `typesite`, `region`, `yc`, `chloc`, `inround`, `panel` and `deceased`.

Seven out of Thirteen tables will be shown in this report. T-SQL statements used to generate the table for `dbo.Child_Identity_and_Location` is shown in Figure 6.1.1

```
/** creating Table for Child_Identity_and_Location */
Create Table dbo.Child_Identity_and_Location(
    [childid] [nvarchar](150) NOT NULL,
    [round] [float] NOT NULL,
    [clustid] [float] NULL,
    [commid] [nvarchar](150) NULL,
    [typesite] [float] NULL,
    [region] [float] NULL,
    [yc] [float] NULL,
    [childloc] [float] NULL,
    [inround] [float] NULL,
    [panel] [float] NULL,
    [deceased] [nvarchar](150) NULL,
    Constraint [PK_Child_Identity_and_Location] Primary Key Clustered
    (
        [childid] asc,
        [round] asc))
Go
```

Figure 6.1.1 creating table for `Child_Identity_and_Location`

The Child anthropometric information table comprises each child's physical characteristics like child's weight (`chweight`), height (`chheight`), body mass index (`bmi`), z-scores for weight for-height (`zwfa`), z-scores for height-for-age (`zhfa`), BMI-for age (`zbfa`), low weight for age (underweight), short height for age (stunting), low BMI for age (thinness) and so on. T-SQL statements used to generate the table for `dbo.Child_anthropometric` is shown in Figure 6.1.2

```

/** creating Table for Child_Anthropometric information */
Create Table dbo.Child_Anthropometric(
    [childid] [nvarchar](150) NOT NULL,
    [round] [float] NOT NULL,
    [chweight] [float] NULL,
    [chheight] [float] NULL,
    [bmi] [float] NULL,
    [zwfa] [float] NULL,
    [zhfa] [float] NULL,
    [zbfa] [float] NULL,
    [zwfl] [nvarchar](150) NULL,
    [fwfa] [float] NULL,
    [fhfa] [float] NULL,
    [fbfa] [float] NULL,
    [fwfl] [nvarchar](150) NULL,
    [underweight] [float] NULL,
    [stunting] [float] NULL,
    [thinness] [float] NULL
) On [PRIMARY]
Go

```

Figure 6.1.2 creating table for Child_Anthropometric

The child parents' features table comprises child's parents' characteristics like father's ID (dadid), father's age (dadage), location of father (dadlive), year the father died (dadyrdied), father cannot read in any language of the countries surveyed (dadcantread), father's level of education (dadedu), mother's ID (momid), mother's level of education (momedu),, location of mother (momlive), mother's age (momage), mother cannot read in any language of the countries surveyed (momcantread) and year the mother died (momyrdied). T-SQL statements used to generate the table for dbo.Child_Parent is shown in Figure 6.1.3

```

/** creating Table for Child_parent features */
Create Table dbo.Child_Parent(
    [childid] [nvarchar](150) NOT NULL,
    [round] [float] NOT NULL,
    [dadid] [nvarchar](150) NULL,
    [dadage] [nvarchar](150) NULL,
    [dadlive] [nvarchar](150) NULL,
    [dadyrdied] [nvarchar](150) NULL,
    [dadcantread] [nvarchar](150) NULL,
    [dadedu] [nvarchar](150) NULL,
    [momid] [float] NULL,
    [momedu] [float] NULL,
    [momlive] [float] NULL,
    [momage] [float] NULL,
    [momcantread] [nvarchar](150) NULL,
    [momyrdied] [nvarchar](150) NULL
) On [PRIMARY]
Go

```

Figure 6.1.3 creating table for Child_Parent

The child birth and immunisation table comprises child's birth weight in grams (bwght), birth weight from documentation (bwdoc), mother's number of antenatal visits during pregnancy with Young Lives child (numante), mother received at least two injections for 13 tetanus during pregnancy with Young Lives child (tetanus), mother was attended by skilled health personnel during delivery (delivery), child has received BCG vaccination (bcg), child has received measles vaccination (measles), child has received polio vaccination (polio), child has received DPT vaccination (dpt) and child has received HIB vaccination (hib). T-SQL statements used to generate the table for dbo.Birth_and_Immunisations is shown in Figure 6.1.4

```

/** creating Table for Child birth and immunisation */
Create Table dbo.Birth_and_Immunisations(
    [childid] [nvarchar](150) NOT NULL,
    [round] [float] NOT NULL,
    [bwght] [nvarchar](150) NULL,
    [bwdoc] [nvarchar](150) NULL,
    [numante] [nvarchar](150) NULL,
    [delivery] [nvarchar](150) NULL,
    [tetanus] [nvarchar](150) NULL,
    [bcg] [nvarchar](150) NULL,
    [measles] [nvarchar](150) NULL,
    [dpt] [nvarchar](150) NULL,
    [polio] [nvarchar](150) NULL,
    [hib] [nvarchar](150) NULL
) On [PRIMARY]
Go

```

Figure 6.1.4 creating table for Birth_and_Immunisations

The child education table comprises child's education characteristics like child has attended pre-primary school (preprim), age at start of Grade 1 (agegr1), child enrolled in formal school during survey year (enrol), grade enrolled during survey year (engrade), type of school enrolled during survey year (entype), highest grade completed at time of interview (hghgrade) and travel time to school in minutes (timesch). T-SQL statements used to generate the table for dbo.Child_Educations is shown in Figure 6.1.5

```

    /*** creating Table for Child_Education ***/
    Create Table dbo.Child_Educatations(
        [childid] [nvarchar](150) NOT NULL,
        [round] [float] NOT NULL,
        [preprim] [float] NULL,
        [agegr1] [float] NULL,
        [enrol] [float] NULL,
        [engrade] [float] NULL,
        [entype] [float] NULL,
        [hghgrade] [float] NULL,
        [timesch] [float] NULL
    ) On [PRIMARY]
    Go

```

Figure 6.1.5 creating table for Child_Educatations

The child read and write table comprises characteristics like child's reading level (levlread), child's writing level (levlwrit), child can read and write a sentence without difficulty (literate). T-SQL statements used to generate the table for dbo. ChildRead_and_Writes is shown in Figure 6.1.6

```

    /*** creating Table for Child read and write features ***/
    Create Table dbo.ChildRead_and_Writes(
        [childid] [nvarchar](150) NOT NULL,
        [round] [float] NOT NULL,
        [levlread] [nvarchar](150) NULL,
        [levlwrit] [nvarchar](150) NULL,
        [literate] [nvarchar](150) NULL
    ) On [PRIMARY]
    Go

```

Figure 6.1.6 creating table for ChildRead_and_Writes

The child subjective health and well-being table comprises characteristics like child's health compared to peers (chhrel), child's health in general (chhealth) and child's subjective well-being using the nine-step ladder (cladder). T-SQL statements used to generate the table for dbo. Child_Reproductive_Health is shown in Figure 6.1.7

```

    /*** creating Table for Child health and wellbeing ***/
    Create Table dbo.Child_Health_and_Wellbeing(
        [childid] [nvarchar](150) NOT NULL,
        [round] [float] NOT NULL,
        [chhrel] [float] NULL,
        [chhealth] [nvarchar](150) NULL,
        [cladder] [nvarchar](150) NULL
    ) On [PRIMARY]
    Go

```

Figure 6.1.7 creating table for Child_Health_and_Wellbeing

6.2 Views

Five views were created but two would be analyzed in this report. Child health history using different cases such as Region, Location, Gender, Child's frequency of smoking, when Child consume alcohol, Child health condition and whether Child might die joining different tables together is shown in figure 6.2.1 below.

```
task1.sql - HP,Child...erty (np,user (bu))  SQLQuery2.sql - not connected task2.sql - n
/**Creating view for child health history **/
Create view dbo.vChild Health History
As
(Select a.childid,a.round,
Region = case /**For region of residence of 2 countries**/
when a.region = 1 then 'Tigray in Ethiopia'
when a.region = 2 then 'Afar in Ethiopia'
when a.region = 3 then 'Amhara in Ethiopia'
when a.region = 4 then 'Oromiya in Ethiopia'
when a.region = 5 then 'Somali in Ethiopia'
when a.region = 6 then 'Benshangul Gumz in Ethiopia'
when a.region = 7 then 'SNNP in Ethiopia'
when a.region = 12 then 'Gambela in Ethiopia'
when a.region = 13 then 'Harari in Ethiopia'
when a.region = 14 then 'Addis Ababa City Administration in Ethiopia'
when a.region = 15 then 'Dire Dawa City Administration in Ethiopia'
when a.region = 24 then 'Others in India'
when a.region = 23 then 'Telangana in India'
when a.region = 77 then 'Not known'
when a.region = 22 then 'Rayalaseema in India'
when a.region = 21 then 'Coastal Andhra in India'
else 'Not Stated'
end,
Location = case /**For area of residence urban/rural**/
when a.typesite = 1 then 'Urban'
when a.typesite = 2 then 'Rural'
when a.typesite = 77 then 'Not known'
end,

Gender = case /**For gender type**/
when f.chsex = 1 then 'Male'
when f.chsex = 2 then 'Female'
else 'Not Stated'
end,
ChildSmokes = case /**Child's frequency of smoking**/
when b.chsmoke = 1 then 'Every day'
when b.chsmoke = 2 then 'At least once a week'
when b.chsmoke = 3 then 'At least once a month'
when b.chsmoke = 4 then 'Hardly ever'
when b.chsmoke = 5 then 'I never smoke cigarettes'
else 'Not Stated'
end,
ChildDrinks = case /**Child consume alcohol everyday or at least once a week**/
when b.chalcohol = 0 then 'No'
when b.chalcohol = 1 then 'Yes'
else 'Not Stated'
end,
ChildHealth = case /**Child's health in general**/
when c.chhealth = 1 then 'Very poor'
when c.chhealth = 2 then 'Poor'
when c.chhealth = 3 then 'Average'
when c.chhealth = 4 then 'Good'
when c.chhealth = 5 then 'Very good'
else 'Not Stated'
end,

ChildMightDie = case /**Child has had serious injury/illness**/
when d.chmightdie = 0 then 'no'
when d.chmightdie = 1 then 'yes'
else 'Not Stated'
end
from dbo.Child_Identity_and_Location a
left join Child_Smokes_and_Drink b on b.childid=a.childid and b.round=a.round
left join Child_Health_and_Wellbeing c on c.childid=a.childid and c.round=a.round
left join ChildInjury d on d.childid=a.childid and d.round=a.round
left join Child_General_Feature f on f.childid=a.childid and f.round=a.round)
Go
Select * From dbo.vChild_Health_History
```

Figure 6.2.1 creating view for child health history

Creating another view for Child, Parent and Carer Education level using different cases such as Region, Location, Gender, Enrollment history, Child Read and Write level, Carer level of education and Parent Level of education by joining different tables together. This is shown in Figure 6.2.2 below.

```
/**Creating a view for child, parent and carer education level**/  
Create view dbo.vGenEducationHistory  
As  
Select a.childid,a.round,  
CountryRegion = case /**For region of residence of 2 countries**/  
when a.region = 1 then 'Tigray in Ethiopia'  
when a.region = 2 then 'Afar in Ethiopia'  
when a.region = 3 then 'Amhara in Ethiopia'  
when a.region = 4 then 'Oromiya in Ethiopia'  
when a.region = 5 then 'Somali in Ethiopia'  
when a.region = 6 then 'Benshangul Gumz in Ethiopia'  
when a.region = 7 then 'SNNP in Ethiopia'  
when a.region = 12 then 'Gambela in Ethiopia'  
when a.region = 13 then 'Harari in Ethiopia'  
when a.region = 14 then 'Addis Ababa City Administration in Ethiopia'  
when a.region = 15 then 'Dire Dawa City Administration in Ethiopia'  
when a.region = 24 then 'Others in India'  
when a.region = 23 then 'Telangana in India'  
when a.region = 77 then 'Not known'  
when a.region = 22 then 'Rayalaseema in India'  
when a.region = 21 then 'Coastal Andhra in India'  
  
else 'Not Stated'  
end,  
[Location] = case /**For area of residence urban/rural**/  
when a.typesite = 1 then 'Urban'  
when a.typesite = 2 then 'Rural'  
when a.typesite = 77 then 'Not known'  
end,  
  
Gender = case /**For gender type**/  
when f.chsex = 1 then 'Male'  
when f.chsex = 2 then 'Female'  
else 'Not Stated'  
end,  
Enrollment = case /**Child is currently enrolled in school**/  
when b.enrol = 0 then 'No'  
when b.enrol = 1 then 'Yes'  
when b.enrol = 99 then 'Missing'  
when b.enrol = 77 then 'Not Known'  
when b.enrol = 88 then 'N/A'  
else 'Not Stated'  
end,  
ChildReadLevel = case /**Child read level**/  
when c.levlread = 1 then 'Cant read anything'  
when c.levlread = 2 then 'Reads letters'  
when c.levlread = 3 then 'Reads word'  
when c.levlread = 4 then 'Reads sentence'  
when c.levlread = 79 then 'Refused to answer'  
else 'Not Stated'  
end,  
ChildWriteLevel = case /**Child write level**/  
when c.levlwrit = 1 then 'No'  
when c.levlwrit = 2 then 'Yes with difficulty or errors'  
when c.levlwrit = 3 then 'Yes without difficulty or errors'  
when c.levlwrit = 79 then 'Refused to answer'  
else 'Not Stated'  
end,
```

```

CarerEducation = case /***Caregiver's level of education***/
when d.caredu = 0 then 'None'
when d.caredu = 1 then 'Grade 1'
when d.caredu = 2 then 'Grade 2'
when d.caredu = 3 then 'Grade 3'
when d.caredu = 4 then 'Grade 4'
when d.caredu = 5 then 'Grade 5'
when d.caredu = 6 then 'Grade 6'
when d.caredu = 7 then 'Grade 7'
when d.caredu = 8 then 'Grade 8'
when d.caredu = 9 then 'Grade 9'
when d.caredu = 10 then 'Grade 10'
when d.caredu = 11 then 'Grade 11'
when d.caredu = 12 then 'Grade 12'
when d.caredu = 13 then 'Post-secondary/vocational'
when d.caredu = 14 then 'University'
when d.caredu = 15 then 'Masters, doctorate'
when d.caredu = 16 then 'University (complete)'
when d.caredu = 28 then 'Adult literacy'
when d.caredu = 29 then 'Religious education'
when d.caredu = 30 then 'Other'
else 'Not Stated'
end,

```

```

FatherEducation = case /***Father's level of education***/
when e.dadedu = 0 then 'None'
when e.dadedu = 1 then 'Grade 1'
when e.dadedu = 2 then 'Grade 2'
when e.dadedu = 3 then 'Grade 3'
when e.dadedu = 4 then 'Grade 4'
when e.dadedu = 5 then 'Grade 5'
when e.dadedu = 6 then 'Grade 6'
when e.dadedu = 7 then 'Grade 7'
when e.dadedu = 8 then 'Grade 8'
when e.dadedu = 9 then 'Grade 9'
when e.dadedu = 10 then 'Grade 10'
when e.dadedu = 11 then 'Grade 11'
when e.dadedu = 12 then 'Grade 12'
when e.dadedu = 13 then '(Post-secondary, vocational/Technical college)/
PE(Technical Incomplete)'
when e.dadedu = 14 then '(University)/
PE(Technical Complete)'
when e.dadedu = 15 then '(Masters, doctorate)/
PE(Uni Incomplete)'
when e.dadedu = 16 then 'PE(Uni Complete)'
when e.dadedu = 28 then 'Adult literacy'
when e.dadedu = 29 then 'Religious education'
when e.dadedu = 30 then 'Other'
else 'Not Stated'
end,

```

```

MotherEducation = case /***Mother's level of education***/
when e.momedu = 0 then 'None'
when e.momedu = 1 then 'Grade 1'
when e.momedu = 2 then 'Grade 2'
when e.momedu = 3 then 'Grade 3'
when e.momedu = 4 then 'Grade 4'
when e.momedu = 5 then 'Grade 5'
when e.momedu = 6 then 'Grade 6'
when e.momedu = 7 then 'Grade 7'
when e.momedu = 8 then 'Grade 8'
when e.momedu = 9 then 'Grade 9'
when e.momedu = 10 then 'Grade 10'
when e.momedu = 11 then 'Grade 11'
when e.momedu = 12 then 'Grade 12'
when e.momedu = 13 then '(Post-secondary, vocational/Technical college)/
PE(Technical Incomplete)'
when e.momedu = 14 then '(University)/
PE(Technical Complete)'
when e.momedu = 15 then '(Masters, doctorate)/
PE(Uni Incomplete)'
when e.momedu = 16 then 'PE(Uni Complete)'
when e.momedu = 28 then 'Adult literacy'
when e.momedu = 29 then 'Religious education'
when e.momedu = 30 then 'Other'
else 'Not Stated'
end

```

```

From Child_Identity_and_Location a
Left join Child_Educations b on b.childid=a.childid and b.round = a.round
Left join ChildRead_and_Writes c on c.childid = a.childid and c.round=a.round
Left join Child_Caregiver_Info d on d.childid = a.childid and d.round=a.round
Left join Child_Parent e on e.childid=a.childid and e.round=a.round
Left join Child_General_Feature f on f.childid=a.childid and f.round=a.round
Go

Select * From dbo.vGenEducationHistory

```

Figure 6.2.2 Creating View for child, Carer and Parent Education History

6.3 Stored Procedure

Stored procedure was created to find child enroll type in various locations also using the group by function also. This is shown in Figure 6.3.1 below.

```
/**Stored Procedures and Groupby**/  
Create Procedure dbo.ChildEnrollment  
@REGION as nvarchar (100),  
@Enrol as nvarchar (10),  
@Sex as nvarchar (10),  
@CurrentEnrol as nvarchar(100),  
@ReadStatus as nvarchar (50),  
@WriteStatus as nvarchar (50)  
As  
Select Gender,[Location],Enrollment,CountryRegion,ChildWriteLevel,ChildReadLevel,  
count (*) As Result From dbo.vEducationAnalysis  
Where CountryRegion=@REGION and Enrollment=@Enrol and Gender=@Sex  
and ChildWriteLevel=@WriteStatus  
Group By Gender,[Location],Enrollment,CountryRegion,ChildWriteLevel,ChildReadLevel
```

Figure 6.3.1 creating stored procedure to find out child enrollment type in different locations

6.4 User Defined Function

A function is created based on the cohort types also using the group by and order by. This is shown in Figure 6.4.1 below.

```
/**Creating functions based on the cohort types**/  
Create function dbo.Cohorts (@youngeroldercohort as bit)  
Returns table  
As  
Return  
(  
Select a.childid, a.round, a.yc, b.CarerEducation,  
Y_C = case  
When a.yc = 0 then 'older'  
When a.yc = 1 then 'younger'  
else null  
end, b.CountryRegion, count (*) As Total  
From Child_Identity_and_Location a left join  
dbo.vEducationAnalysis b on b.childid=a.childid and b.round=a.round  
Where yc = @youngeroldercohort  
Group by a.childid,a.round,b.CountryRegion, b.CarerEducation,a.yc  
Order by childid, round offset 0 rows  
)  
Go  
Select * From dbo.Cohorts(1)  
Select * From dbo.Cohorts(0)
```

Figure 6.4.1 creating function based on cohort types

Count function was also used to count the total number of residents in different areas, then group them by their type site. This is shown in Figure 6.4.2 below

```
/**counts the total number of residents in different areas, then groups them by their typesite**/  
Select count(*) as 'Total Number of Residents', typesite  
from dbo.india_constructed  
group by typesite;  
  
/**counts the total number of residents in different areas, then groups them by their typesite**/  
Select count(*) as 'Total Number of Residents', typesite  
from dbo.ethiopia_constructed  
group by typesite;
```

Figure 6.4.2 Using Count functions to count the total number of residents in different areas, then group them by their type site

8. Database Security

Enforcing security is very crucial in the design of a database design. Two roles were created using the select, insert and update access for the HODs in Education and Health unit. This is shown in Figure 8.1 below

```
/**Enforcing security for HOD of departments**/  
/**Education HOD only**/  
Create role EducationHOD  
Grant select, insert, update on dbo.vGenEducationHistory to  
EducationHOD  
  
/**Health HOD only**/  
Create role HealthHOD  
Grant select, insert, update on [dbo].[vChild_Health_History] to  
HealthHOD
```

Figure 8.1 Enforcing security for Education and Health HODS

9. Database Backup and Restore Strategy

It is guided to test your strategy by restoring a set of backups and then recovering your database to prepare you to respond effectively to a disaster. For the Child_Poverty database, a full backup and restore was created in the computer C: Drive was done daily from time to time while working on the database. This is shown in Figure 9.1 below

```
/**creating Full Backup Database**/  
Backup Database Child_Poverty  
To Disk = 'C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\Child_Poverty.bak'  
Go  
  
/** Restoring Database**/  
Use [master]  
Restore Database Child_Poverty  
From Disk = 'C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\Child_Poverty.bak'  
Go
```

Figure 9.1 Creating Full Backup and Restore Database

10. Data Science/Business Intelligence Techniques

Data Science technique for this task on child poverty index would be clustering. This method can be used to cluster cohorts interviewed by characteristics which are similar to them like location, age, sex, school type, level of education and so on. The Schema diagram created can be named as a form of Business Intelligence tool to show the relationship between tables, views or variables in the database.

11. Data Privacy, Ethical and legal issues

The datasets used for this report were acquired from the UK data service and sufficient assessment of gaining consent and anonymizing data was applied.

12. Conclusion

The objective of this task is to create and design a database system that gives report to measure the child poverty in Ethiopia and India but we didn't use a detailed data science methods and algorithms to create the database and generate the reports.

TASK 2

The design and Implementation using T- SQL statement for various summarized reports for the inequality in the education sector in Vietnam.

1. Abstract

Datasets used comprises school survey data for Vietnam which was conducted between 2016 and 2017. The survey covers more of school and education related questions. The objective is to develop a reporting tool to enable us compare the variables that has contributed to the inequality in the education sector in Vietnam. Tables, views, stored procedure were created in SQL Server and various summarized report was created in excel to help achieve this.

2. Introduction

Dataset used for this task is a gathering of a survey report gotten from a research carried out by the Young Lives project and it entails school survey data conducted in Vietnam between 2016 to 2017. The report comprises the result of the significant questions asked to the children, results of their cognitive tests, class roster and information gathered from their Head teacher/Principal in Vietnam. The objective is to develop a reporting tool to enable us compare the variables that has contributed to the inequality in the education sector in Vietnam. The database is named “Vietnamdb”. Vietnam_wave_1 dataset is inserted into the Child_Poverty database. The inbuilt schema is “dbo”. The table is are created on the MS SQL Server with the schema name dbo. vietnam_wave_1.

4. Design Rationale

Design rationale is the same with task 1 above.

5. Design Considerations

Design consideration is the same with task 1 above. No constraints was used.

6. T-SQL Statements

6.1 Tables

Seven tables were created from the datasets using the metadata and data dictionaries. For this task, only the `EduIneq.vietnam_wave_1` was used. The tables were classified based on the different child characteristics and questions asked in the survey.

The first table is classified based on the child's identifiers and location. This table is named `StudentIdentifiers`. The variables in this table have been used to uniquely identify students (`UNIQUEID`), school (`SCHOOLID`), section (`CLASSID`), districts where schools are located (`DISTRICTCODE`), and the Young Lives sample sites where they are located (`LOCALITY`). The table for `StudentIdentifiers` is shown in Figure 6.1.1 below.

```
/**Creating table for Student Identifiers**/  
Select UNIQUEID,SCHOOLID,CLASSID,  
STUDENTID,YLCHILDID,PROVINCE,DISTRICTCODE,LOCALITY  
Into dbo.StudentIdentifiers  
From dbo.vietnam_wave_1  
  
Select * From dbo.StudentIdentifiers
```

Figure 6.1.1 Creating table for StudentIdentifier

The `Student Questionnaire` table contains each child's characteristics such as child's unique ID, student ID, gender, age, ethnicity, days absent from school (`ABSENT_DAYS`), can mom read and write (`MOM_READ`), mom's level of education (`MOM_EDUC`), can dad read and write (`DAD_READ`), dad's level of education (`DAD_EDUC`), is student a YL Child (`STDYLCHD`) and date of interview (`STDDINT`). The table for `StudentQuestionnaire` is shown in Figure 6.1.2 below.

```
/**Creating table for Student Questionnaire**/  
Select UNIQUEID,STUDENTID,GENDER,AGE  
,ETHNICITY,ABSENT_DAYS,MOM_READ,MOM_EDUC,DAD_READ  
,DAD_EDUC,STDYLCHD,STDDINT  
Into dbo.StudentQuestionnaire  
From dbo.vietnam_wave_1  
  
Select * From dbo.StudentQuestionnaire
```

Figure 6.1.2 Creating table for StudentQuestionnaire

The Student Cognitive tests table contains characteristics such as child's student ID, attempted English test (ENG_TEST), attempted Maths test (MATH_TEST), scored English test (ENG_RAWSCORE) and scored Maths test (Math_Rawscore). The table for StudentCognitivetests is shown in Figure 6.1.3 below.

```
/**Creating table for Student Cognitive tests**/  
Select STUDENTID,ENG_TEST,Math_Rawscore,  
ENG_RAWSCORE, MATH_TEST  
Into dbo.StudentCognitivetests  
From dbo.vietnam_wave_1  
  
Select * From dbo.StudentCognitivetests
```

Figure 6.1.3 Creating table for StudentCognitivetests

The Student Gender table contains the students gender type with the associated figure used in depicting it in the survey. The table for StudentGender is shown in Figure 6.1.4 below.

```
/**Creating table for Student Gender**/  
Create Table dbo.StudentGender(  
GENDER int,  
GENDER_TYPE nvarchar(8) not null  
);  
  
/**Inserting Into the Gender table**/  
Insert Into dbo.StudentGender(GENDER , GENDER_TYPE)  
values(1, 'Male'),  
      (2, 'Female')
```

Figure 6.1.4 Creating and Inserting into StudentGender table

The Student Ethnicity table comprises the student's ethnicity ID and ethnicity name. The table for StudentEthnicity is shown in Figure 6.1.5 below.

```
/**Creating table for the Ethnicity Identity**/  
Create Table dbo.StudentEthnicity(  
ETHNICITYID int,  
ETHNICITY_NAME nvarchar(10)  
);  
  
/**Inserting Into the Ethnicity table**/  
Insert Into dbo.StudentEthnicity(ETHNICITYID , ETHNICITY_NAME)  
values(1, 'Kinh'),  
      (2, 'H'Mong'),  
      (3, 'Cham-HRoi'),  
      (4, 'Ede'),  
      (5, 'Ba Na'),  
      (6, 'Nung'),  
      (7, 'Tay'),  
      (8, 'Dao'),  
      (9, 'Giay'),  
      (10, 'Other')  
  
Select * From dbo.StudentEthnicity
```

Figure 6.1.5 Creating and Inserting into StudentEthnicity table

The Student Locality table contains the student's locality ID and locality type. The table for StudentLocality is shown in Figure 6.1.6 below.

```
/**Creating table for Student Locality types**/  
Create Table dbo.StudentLocality(  
LOCALITYID int,  
LOCALITY_TYPE nvarchar(10) not null  
);  
  
/**Inserting Into the Student Locality table**/  
Insert Into dbo.StudentLocality(LOCALITYID , LOCALITY_TYPE)  
values(1, 'Rural'),  
      (2, 'Urban')  
  
Select * From dbo.StudentLocality
```

Figure 6.1.6 Creating and Inserting into StudentLocality table

The Student Province table contains the student's province and province name. The table for StudentProvince is shown in Figure 6.1.7 below.

```
/**Creating a table Student Province**/  
Create Table dbo.StudentProvince(  
PROVINCE int,  
PROVINCE_NAME nvarchar(10) not null  
);  
  
/**Inserting Into the Student Province table**/  
Insert Into dbo.StudentProvince(PROVINCE , PROVINCE_NAME)  
values(1, 'Ben Tre'),  
      (2, 'Da Nang'),  
      (3, 'Hung Yen'),  
      (4, 'Lao Cai'),  
      (5, 'Phu Yen')  
  
Select * From dbo.StudentProvince
```

Figure 6.1.7 Creating and Inserting into StudentProvince table

6.2 Views

Four views were created but three would be analyzed in this report. Creating a view for Inequality in the Education sector in Vietnam using different cases such as Gender, Ethnicity, Province, locality, Student that have study chair, lamp, school bag, ruler, Calculator, Computer, Internet, School have books, IT facilities, Teacher have desk, Chair and type of school. This is shown in figure 6.2.1 below.

```

/**Creating a view for Inequality_in_the_Education_Sector_in_Vietnamfrom vietnam_wave_1 table***/
Create View dbo.vInequality_in_the_Education_Sector
as

```

```

Select uniqueid,
Case gender
When 1.0 then 'Male'
else 'Female'
end as 'Gender',
Case ethnicity
When 1.0 then 'Kinh'
When 2.0 then 'Hmong'
When 3.0 then 'cham-HRoi'
When 4.0 then 'Ede'
When 5.0 then 'Ba na'
When 6.0 then 'Nung'
When 7.0 then 'Tay'
When 8.0 then 'Dao'
When 9.0 then 'Giay'
else 'Others'
end as 'Ethnicity',
Case province
When 1.0 then 'Ben Tre'
When 2.0 then 'Dan Nang'
When 3.0 then 'Hung Yen'
When 4.0 then 'Lao Cai'
When 5.0 then 'Phu Yen'
else 'Unknown'
end as 'Province',

```

```

Case locality
When 1.0 then 'Rural'
When 2.0 then 'Urban'
else 'Unknown'
end as 'School Location',
Case sthvchr
When 0.0 then 'No'
else 'Yes'
end as 'Have Study Chair at Home',
Case sthvlamp
When 0.0 then 'No'
else 'Yes'
end as 'Have Study Lamp at Home',
Case stitmow5
When 0.0 then 'No'
else 'Yes'
end as 'Student has School Bag',
Case stitmow6
When 0.0 then 'No'
else 'yes'
end as 'Student has Ruler',
Case stitmow8
When 0.0 then 'No'
else 'yes'
end as 'Student has Pocket Calculator',
Case sthvcomp
When 0.0 then 'No'
else 'yes'
end as 'Have Computer/Laptop at Home',

```



```

Case sthvintr
When 0.0 then 'No'
else 'yes'
end as 'Have Internet at Home',
Case scavlb5
When 0.0 then 'No'
else 'Yes'
end as 'School have Books',
Case scavlb10
When 0.0 then 'No'
else 'yes'
end as 'School have IT Facilities',
Case scavlb3
When 0.0 then 'No'
else 'Yes'
end as 'Teacher has desk',
Case scavlb4
When 0.0 then 'No'
else 'Yes'
end as 'Teacher has chair',
Case httpsch
When 1.0 then 'Government'
When 2.0 then 'Private'
When 3.0 then 'Other'
else 'Unknown'
end as 'Type of School'
from dbo.vietnam_wave_1

Select * From dbo.vInequality_in_the_Education_Sector

```

Figure 6.2.1 Creating view for Inequality in the education sector in Vietnam

Creating another one for viewing the number of province using group by and count function. This is shown in Figure 6.2.2 below.

```

/**Viewing the number of Province using group by and count***/
create view dbo.vprovince
as
Select PROVINCE_NAME As Province, (Count(PROVINCE_NAME)) As Number_of_Province
From dbo.vStudentIdentifiers
Group By PROVINCE_NAME

select * from dbo.vprovince

```

Figure 6.2.2 Creating view for counting the number of province

Creating another one for the Student Grades. This is shown in Figure 6.2.3 below

```

/**View for Student Grades***/
Create View dbo.vGrade As
Select b.SCHOOLID,b.STUDENTID,b.CLASSID,
c.Math_Rawscore As MATH_RAWSCORE,c.ENG_RAWSCORE
From
dbo.vStudentIdentifiers b,
dbo.StudentCognitivetests c

select * from dbo.vGrade

```

Figure 6.2.3 Creating view for Student Grades

6.3 Stored Procedure

Stored procedure was created for finding student province. This is shown in Figure 6.3.1 below.

```
/**Stored Procedure for finding Student Province**/  
Create Procedure dbo.ProvinceFinder  
@PROVINCE_NAME nvarchar (30) = null  
As  
Select UNIQUEID,PROVINCE_NAME From dbo.vStudentIdentifiers Where isnull  
(@PROVINCE_NAME,PROVINCE_NAME) = PROVINCE_NAME  
  
Execute dbo.ProvinceFinder @PROVINCE_NAME ='Hung Yen'
```

Figure 6.3.1 Stored Procedure to find the student province

7. Report Design

For this task, the views were made to represent a subset of the vietnam_wave_1 dataset to allow us determine and compare the variables that has contributed to the inequality in the education sector in Vietnam. The visualization tools used in designing the front-end is Microsoft Excel. This is shown in the Figure 7.1 and 7.2 below.

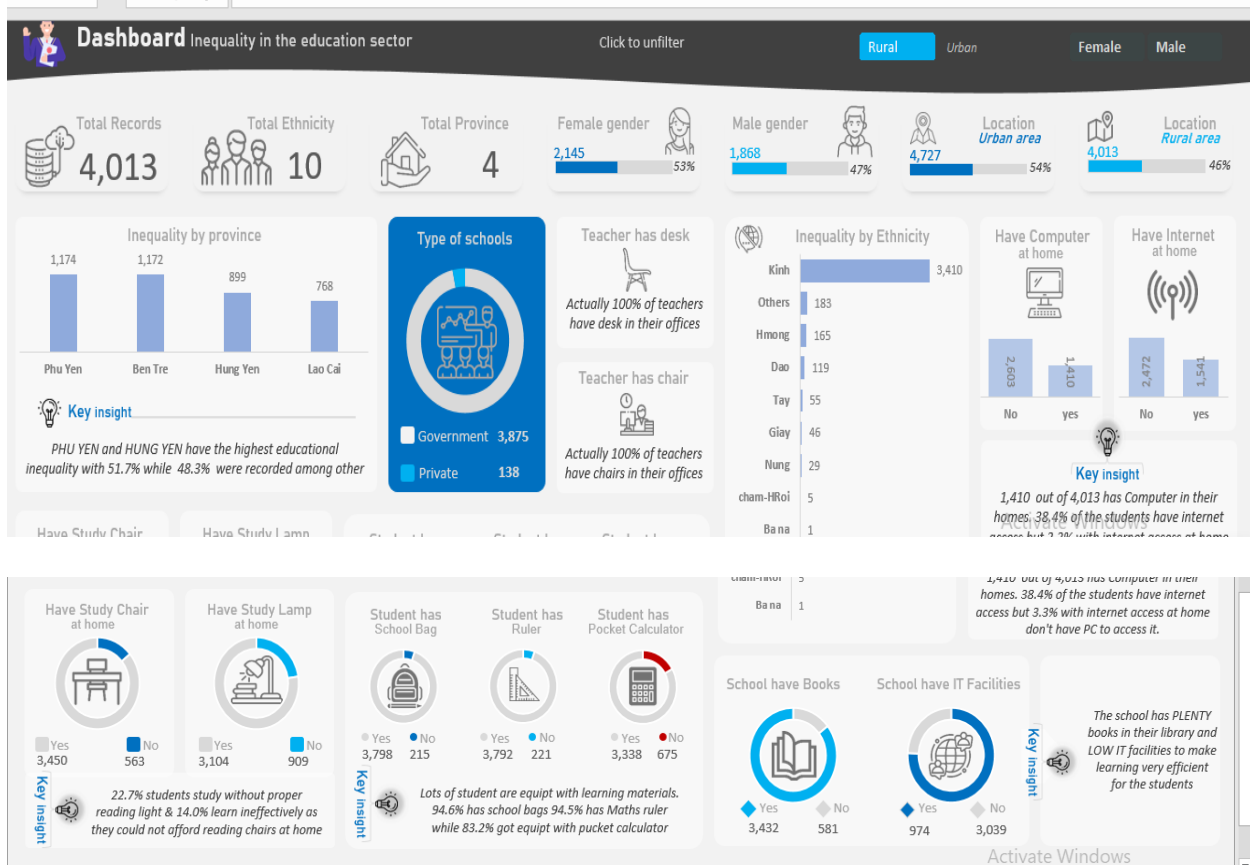


Figure 7.1 Inequality in the Rural Area



Figure 7.2 Inequality in the Urban Area

If Students have Study Chair and Lamp at home

Holistically, 22.4% study without proper reading light and 13.2% learn could not afford reading chairs at home. In the rural area 22.7% do not have reading light and 14% could not afford reading chairs at to learn effectively. The Urban area has 22.3% of student that could not afford proper reading light and 12.5% has no reading chairs at home. Base on the full insight we got from if students have reading chairs or reading light, we found out that greater numbers of students have chairs and reading lights at home but some parts of the students were not fortunate.

If Students have School Bag, Ruler and Pocket Calculator

From the analysis collected from the dataset we found that greater percentage of students with school bag, Ruler and Pocket calculator. 96% has school bags, 94% has ruler and 84% has pocket calculator. Only

few of the students are unfortunate to get equipped with the following: school bag, Ruler and Pocket calculator.

If Teachers have Desk and Chairs

From the analysis we found out that 100% of the teachers have desk and chairs in their offices.

If Students have computer or not

Analysis shows that both rural and urban area have 44.7% of people have internet access at home but 3% of those with internet access do not have Computer to access the available internet. We have more people in the urban area with internet access than that of the rural area.

If School have books and IT facilities

Without filters, there are plenty books in the library and low IT facilities. The rural area has more books than that of the urban area.

8. Database Backup and Restore Strategy

It is guided to test your strategy by restoring a set of backups and then recovering your database to prepare you to respond effectively to a disaster. For the Vietnamdb database, a full backup and restore was created in the computer C: Drive was done daily from time to time while working on the database. This is shown in Figure 8.1 below

```

    /**creating Full Backup Database***/
Backup Database Vietnamdb
To Disk = 'C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\Vietnamdb.bak'
Go

/** Restoring Database***/
Use [master]
Restore Database Vietnamdb
From Disk = 'C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\Vietnamdb.bak'
Go
```

Figure 8.1 Creating Backup and Restore Database for Vietnamdb

11. Conclusion

It is concluded that majority of the students from the rural areas commonly attend government owned schools from the visualization of the students. The government owned schools are extremely under-equipped with no internet facilities. Policy and decision makers should improve the level of facilities and school improvement in rural areas of Vietnam.

TASK 3

The design and Implementation using T- SQL statement for various summarized reports to develop a reporting tool called Crime Profiler and to build a Lower Layer Super Output Areas (LSOAs) wise crime report with local population data in Greater Manchester between January 2017 and December 2018.

1. ABSTRACT

For Task 3, Dataset used is a collection of monthly crime report in Greater Manchester from January 2017 to December 2018 and the LSOA population data within this period. The objective is to develop a reporting tool to enable us create a summarized LSOAs wise crime report with local population data in Greater Manchester between Jan 2017 and Dec 2018. Tables, views, stored procedure in SQL Server and various summarized report was created in excel to help achieve this. The QGIS (Geography Information System) was also used for the analysis of the report.

2. Introduction

Datasets used for this task comprises a collection of monthly crime report in Greater Manchester from January 2017 to December 2018 and the LSOA population data within this period. The objective is to develop a reporting tool called Crime Profiler to assist in the development of computational criminology as a part of the advancement of the theory and method in criminological research and to build a Lower Layer Super Output Areas (LSOAs) wise crime report with local population data in Greater Manchester between January 2017 and December 2018. The datasets for the months considered were downloaded

from the UK Police Data and the UK office for national statistics websites. On the Microsoft SQL Server, a database is created. The database is named “ManchesterCrime”. All the reports in Greater Manchester from January to December 2018 and LSOA population data are inserted into the ManchesterCrime database. The inbuilt schema is “dbo”. The tables are created on the MS SQL Server with the schema name.

4. Design Rationale

Design rationale is the same with task 1 above.

5. Design Considerations

Design consideration is the same with task 1 above.

Constraints

Primary key was used for this task on table dbo.All_Crimes after joining all the crime data in preparation for creating a front end presentation layer for the crime reporting system in Greater Manchester. This is shown in Figure 5.1 below.

```
/**Adding a Geolocation column to the all crime table**/  
Alter Table dbo.All_Crimes  
Add [Geolocation] Geography;  
  
/**Adding an ID column to the table and setting as primary key**/  
Alter Table dbo.All_Crimes  
Add ID Int Identity;  
  
/**Setting a primary key to the added ID column**/  
Alter Table dbo.All_Crimes  
Add Constraint PK_ID Primary Key Nonclustered (ID);  
Go  
  
/**Creating a geography point using Latitude and Longitude columns**/  
Update dbo.All_Crimes  
Set [GeoLocation] = Geography::Point([Latitude], [Longitude], 4326)  
Where [Longitude] IS NOT NULL and [Latitude] IS NOT NULL  
Go
```

6. T-SQL Statements

6.1 Tables

The first table was created by joining all the crime data from January 2017 and December 2018. This table is named 'dbo.All_Crimes'. T-SQL statements used to create the tables for this task are shown in Figure 6.1.1 below.

```

/**Creating and joining all the crime data from 01-2017 to 12-2018 using union all**/
Select * into dbo.All_Crimes
From [dbo].['2017-01-greater-manchester-stre$'] Union All
Select * From [dbo].['2017-02-greater-manchester-stre$'] Union All
Select * From [dbo].['2017-03-greater-manchester-stre$'] Union All
Select * From [dbo].['2017-04-greater-manchester-stre$'] Union All
Select * From [dbo].['2017-05-greater-manchester-stre$'] Union All
Select * From [dbo].['2017-06-greater-manchester-stre$'] Union All
Select * From [dbo].['2017-07-greater-manchester-stre$'] Union All
Select * From [dbo].['2017-08-greater-manchester-stre$'] Union All
Select * From [dbo].['2017-09-greater-manchester-stre$'] Union All
Select * From [dbo].['2017-10-greater-manchester-stre$'] Union All
Select * From [dbo].['2017-11-gloucestershire-street$'] Union All
Select * From [dbo].['2017-12-greater-manchester-stre$'] Union All
Select * From [dbo].['2018-01-gloucestershire-street$'] Union All
Select * From [dbo].['2018-02-greater-manchester-stre$'] Union All
Select * From [dbo].['2018-03-greater-manchester-stre$'] Union All
Select * From [dbo].['2018-04-gloucestershire-street$'] Union All
Select * From [dbo].['2018-05-gloucestershire-street$'] Union All
Select * From [dbo].['2018-06-greater-manchester-stre$'] Union All
Select * From [dbo].['2018-07-greater-manchester-stre$'] Union All
Select * From [dbo].['2018-08-greater-manchester-stre$'] Union All
Select * From [dbo].['2018-09-greater-manchester-stre$'] Union All
Select * From [dbo].['2018-10-greater-manchester-stre$'] Union All
Select * From [dbo].['2018-11-greater-manchester-stre$'] Union All
Select * From [dbo].['2018-12-greater-manchester-stre$'];

/**Viewing the joined table created**/
Select * From dbo.All_Crimes

```

Figure 6.1.1 Joining all the tables to create All_Crimes table

LSOA population data named "mid person\$" was inserted into the SQL Server and joined to the dbo.All_Crimes table. This is shown in Figure 6.1.2 below

```

/**Creating a new table using join**/
Select a.[ID], a.[Crime ID], a.[Month], a.[Reported by], a.[Falls within], a.[Longitude], a.[Latitude],
a.[Location], a.[Geolocation], a.[Crime type], a.[LSOA name],a.[Last outcome category], a.[Context], b.[All Ages]
Into dbo.New_Crimee
From dbo.All_Crimes a
join [dbo].['mid person$'] b on b.[Area Codes]= a.[LSOA code]

Select * From [dbo].['mid person$']
Select * From dbo.New_Crimee

```

Figure 6.1.2 Creating new table New_Crimee by joining LSOA population data

6.2 Views

Views were created by choosing the different crime types. According to Lab 8, the views were created with schemabinding. This is shown in figure 6.2.1 and 6.2.2 below


```

/**View for Robbery in greater manchester**/
Create View dbo.vRobbery
with schemabinding
As
Select [ID], [Crime ID],Location, Geolocation,[Crime type],[LSOA name] From dbo.New_Crimee
Where [Crime type] = 'Robbery'
And [Geolocation] Is not Null

Select * From dbo.vRobbery

/**view to visualise the Anti-Social Behaviour in Salford**/
Create View dbo.vAnti_behaviour_Salford
with schemabinding
As
Select [ID], [Crime ID],Location, Geolocation,[Crime type],[LSOA name] From dbo.New_Crimee
Where [Crime type] = 'Anti-social behaviour' and [LSOA name] like 'Salford%'
And [Geolocation] Is not Null

Select * From  dbo.vAnti_behaviour_Salford

/** view for Vehicle Crime in greater manchester **/
Create View dbo.vVehicle_Crime
with schemabinding
As
Select [ID],[Crime ID],Location, Geolocation,[Crime type],[LSOA name] From dbo.New_Crimee
Where [Crime type] = 'Vehicle Crime'
And [Geolocation] Is not Null

Select * From dbo.vVehicle_Crime

```

Figure 6.2.1 Views created by different Crime types

```

/**View for Public order in greater manchester**/
Create View dbo.vPublic_Order
with schemabinding
As
Select [ID], [Crime ID],Location, Geolocation,[Crime type],[LSOA name] From dbo.New_Crimee
Where [Crime type] = 'Public order'
And [Geolocation] Is not Null

Select * From dbo.vPublic_Order

/**View for Possession of weapons**/
Create View dbo.vPossession_Of_Weapons
with schemabinding
As
Select [ID], [Crime ID],Location, Geolocation,[Crime type],[LSOA name] From dbo.New_Crimee
Where [Crime type] = 'Possession of weapons'
And [Geolocation] Is not Null

Select * From dbo.vPossession_Of_Weapons

```

Figure 6.2.2 Views created by different Crime types

Indexes was also created for the views. This is shown in Figure 6.2.3 below

```

/**Create indexes for all the newly created Views**/
CREATE UNIQUE CLUSTERED INDEX idx_id ON dbo.vRobbery(ID)
GO
CREATE UNIQUE CLUSTERED INDEX idx_id ON dbo.vAnti_behaviour_Salford(ID)
GO
CREATE UNIQUE CLUSTERED INDEX idx_id ON dbo.vVehicle_Crime(ID)
GO
CREATE UNIQUE CLUSTERED INDEX idx_id ON dbo.vPublic_Order(ID)
GO
CREATE UNIQUE CLUSTERED INDEX idx_id ON dbo.vPossession_Of_Weapons(ID)
GO

```

Figure 6.2.3 Creating Indexes for the Views

6.3 Stored Procedure

Stored procedure was created centered on counting the total number of crimes in each month of the review period from January 2017 to December 2018. This is shown in Figure 6.3.1 and Figure 6.3.2 below.

```

/**Creating a stored procedure to count crime type committed in 2017**/
Create Procedure dbo.Crime_2017
@crime NVARCHAR(100)
as
Select [Month] as months,
Count
(Case When [Crime type] = @crime then 1
end) CrimeCount
From dbo.New_Crimee
Where [Crime type] = @crime AND [Month] like '%2017%'
Group By [Month]
Order By [Month]
Go

/**Executing the procedure**/
Execute dbo.Crime_2017 @crime = 'Possession of weapons'

```

Figure 6.3.1 creating stores procedure to count crime type (Possession of weapons) committed in 2017.

```

/**Creating a stored procedure to count crime type committed in 2018**/
Create Procedure dbo.Crime_2018
@crime NVARCHAR(100)
as
Select [Month] as months,
Count
(Case When [Crime type] = @crime then 1
end) CrimeCount
From dbo.New_Crimee
Where [Crime type] = @crime AND [Month] like '%2018%'
Group By [Month]
Order By [Month]
Go

/**Executing the procedure**/
Execute dbo.Crime_2018 @crime = 'Vehicle Crime'

```

Figure 6.3.1 creating stores procedure to count crime type (Vehicle Crime) committed in 2018.

7. Report Design

For this task, the views and reports were made to represent various crime type committed in Greater Manchester in the review period. The visualization tools used in designing the front-end is Microsoft Excel and QGIS.

Anti social behavior in Salford

9.4% of Anti-social behavior happened in those spot which is the highest crime count. This location has crime ranging between 137 to 53. On or near Parking Area, On or near Shopping Area, On or near Sports/Recreation Area, On or near Petrol Station, On or near Pedestrian Subway, On or near Church Street, On or near Police Station, On or near A5185, On or near Supermarket, On or near Cross Lane.

There are very low crime counts in the other crime spot like On or near Brentwood Road, On or near Brereton Grove and On or near A57 etc. Those mention cities has one Anti-social behavior crime each. This is shown in Figure 7.1 below.

Possession of weapon in Greater Manchester.

Out of 2,809 possession of weapon crime spot, these 10 spots have the highest weapon cases. The places are: On or near Parking Area, On or near Airport/Airfield, On or near Shopping Area, On or near Supermarket, On or near Petrol Station, On or near Sports/Recreation Area, On or near Piccadilly, On or near Pedestrian Subway, On or near Nightclub, On or near International Approach. 17% of the crime happened in these spots. Manchester 053D is one of the highest crime location where more weapons were illegally owned. This is shown in Figure 7.1 below.

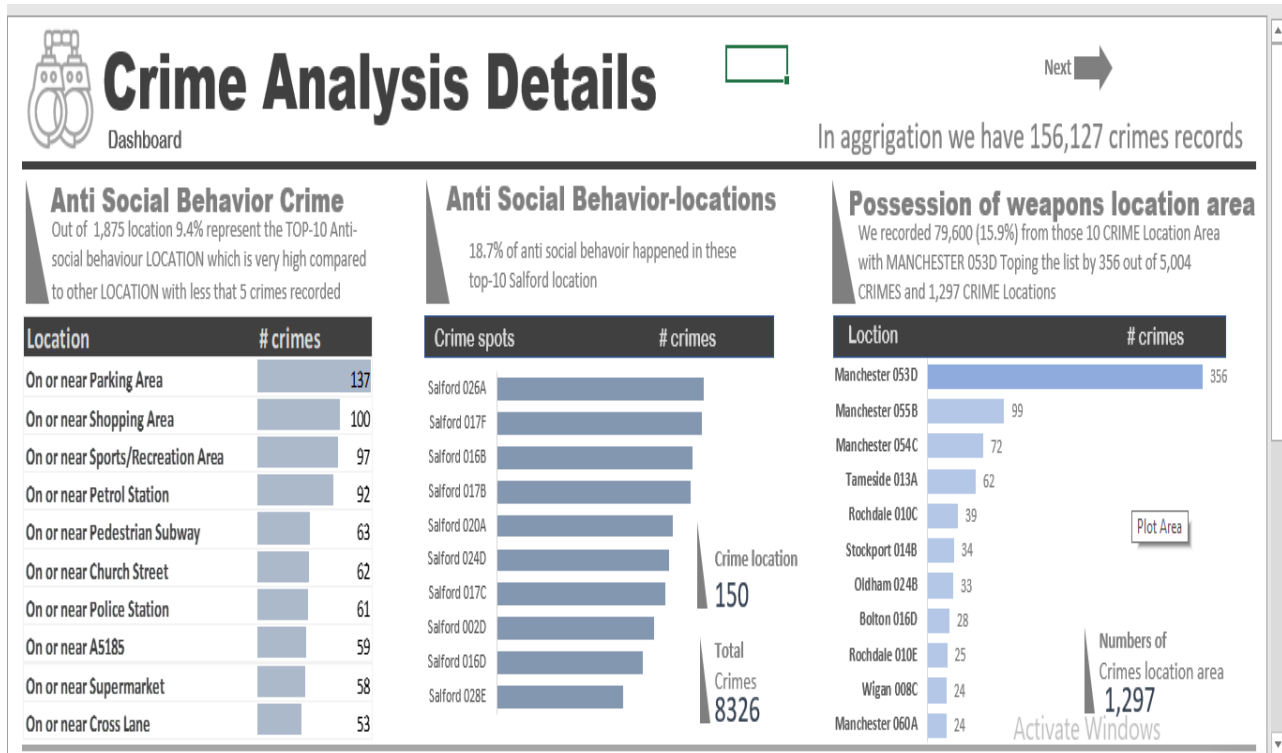


Figure 7.1 Visual Report of Anti-Social Behavior and Possession of Weapons Crime type

Vehicle Crime

On or near Parking Area has the highest number of vehicle crime with over 1,700 records according to our data. Petrol station come second. We recorded more vehicle crime in Manchester 054C location than other locations. This is shown in Figure 7.2 below

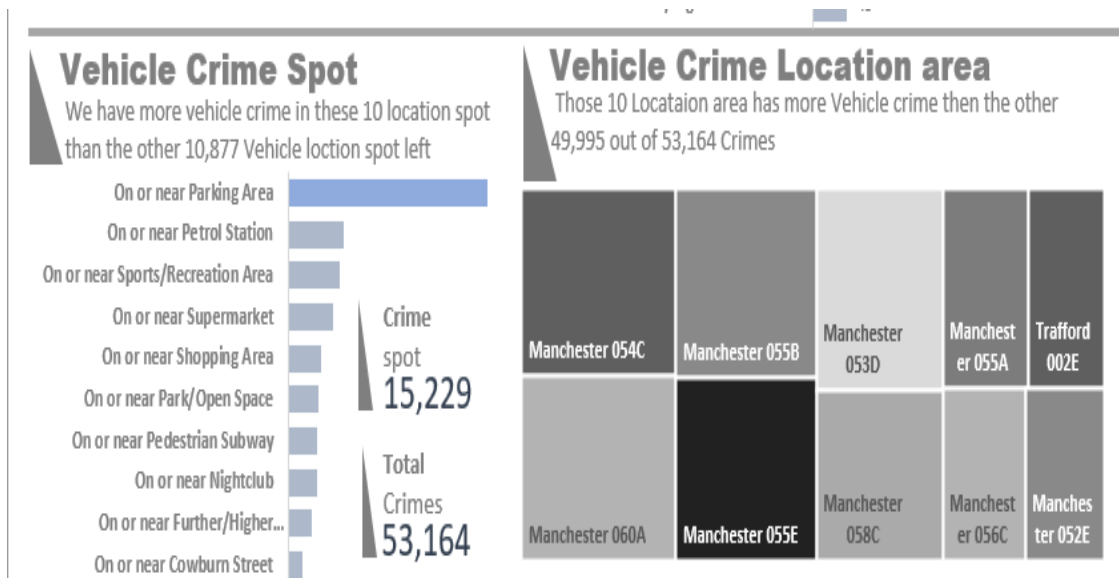


Figure 7.2 Visual report for Vehicle Crime in Greater Manchester

Public Order

Based on our data, we see most of these crime type to be very prevalent in the spot like: On or near parking area, supermarket, shopping areas etc. Based on the analysis Manchester 055B is the location with the highest count. This is shown in Figure 7.3 below.

Robbery crime

We have: On or near Parking Area, On or near Shopping Area, On or near Supermarket, On or near Piccadilly, On or near Nightclub, On or near Sports/Recreation Area, On or near Petrol Station. In the dataset the above spots have robbery ranging from 429 to 157. The other cities has less than 80 to 1 crime per crime spots. Manchester 055B is still the top location where these crime happened. This is shown in Figure 7.3 below.

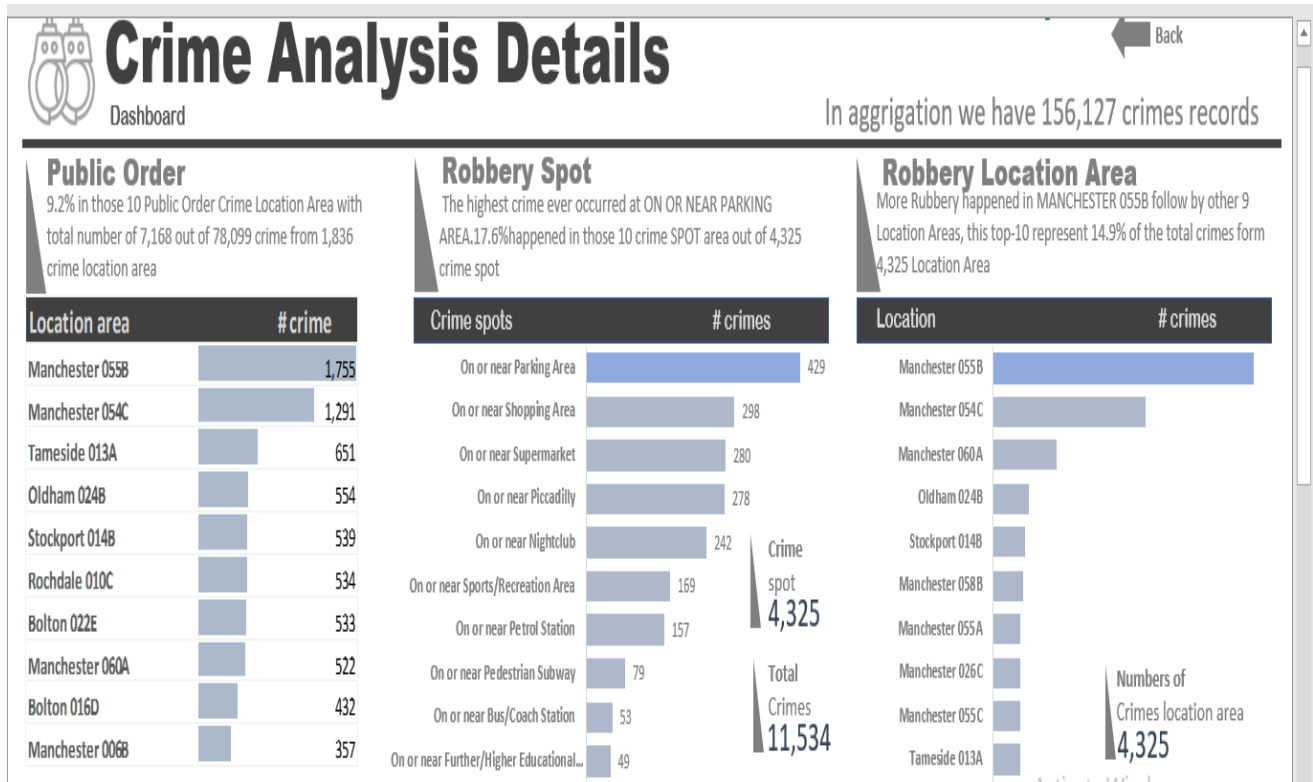


Figure 7.3 Visual report for Public Order and Robbery Crime in Greater Manchester



Figure 7.4 Summary of Five Crimes

Geospatial Map Spread for Anti_Social_Behaviour Crimes in Salford using Google Satellite. This is shown in Figure 7.5 below

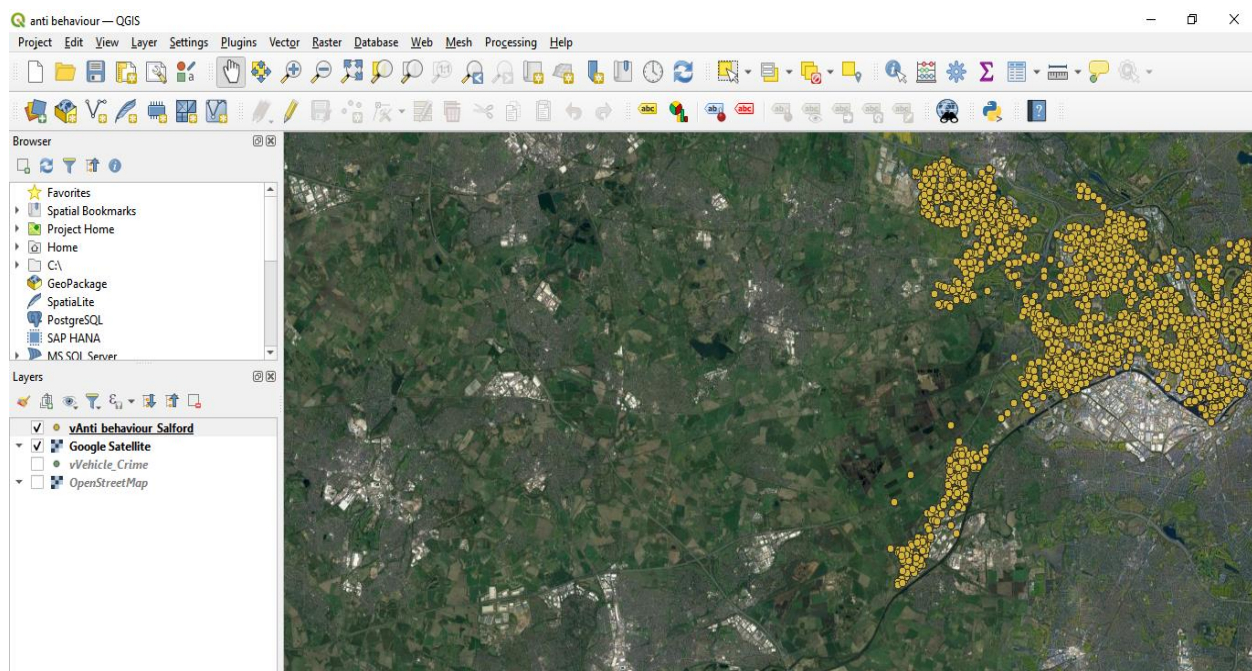


Figure 7.5 Geospatial Map Spread for Anti_Social_Behaviour Crimes in Salford using Google Satellite

Geospatial Map Spread for Vehicle_Crime in Greater Manchester using OpenStreetMap. This is shown in Figure 7.6 below

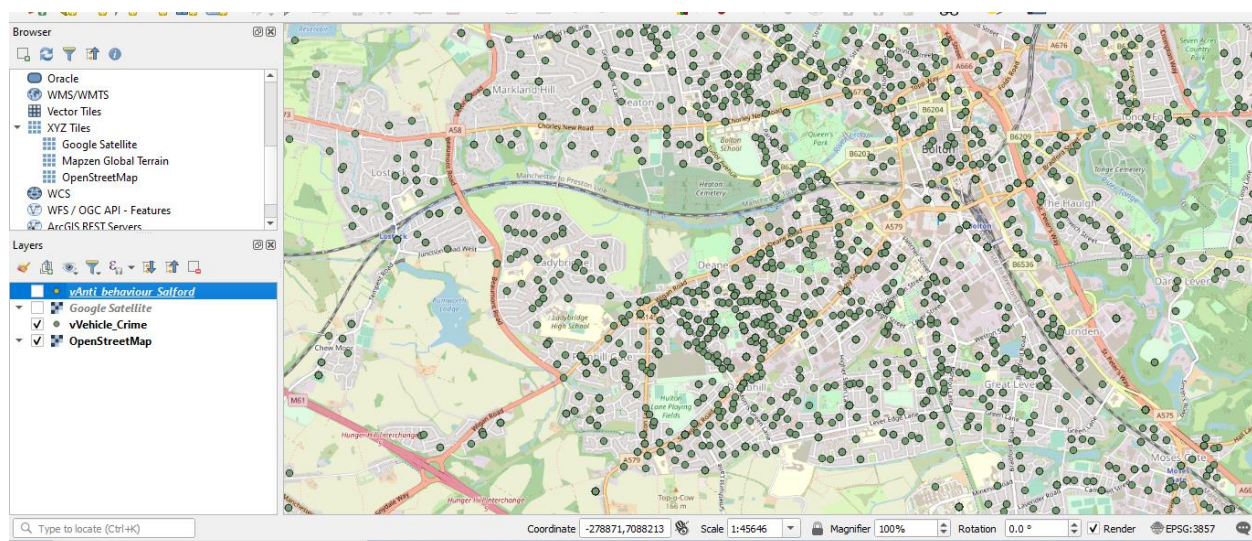


Figure 7.6 Vehicle_Crime in Greater Manchester using OpenStreetMap

8. Database Backup and Restore Strategy

It is guided to test your strategy by restoring a set of backups and then recovering your database to prepare you to respond effectively to a disaster. For the ManchesterCrime database, a full backup and restore was created in the computer C: Drive was done daily from time to time while working on the database. This is shown in Figure 8.1 below

```
/**creating Full Backup Database**/  
Backup Database ManchesterCrime  
To Disk = 'C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\ManchesterCrime.bak'  
Go  
  
/** Restoring Database**/  
Use [master]  
Restore Database ManchesterCrime  
From Disk = 'C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\ManchesterCrime.bak'  
Go
```

Figure 8.1 Creating Back Up and Restoring Database for ManchesterCrime.

10. Data Science/Business Intelligence Techniques

The use of visualizations with QGIS and Microsoft Excel are business intelligence techniques used to build the front end design. Data normalization was done as a data science technique.

12. Conclusion

It is concluded based on the insight from our data we have more crimes coming from on or near parking spot than any other crime spots while Manchester 053D is one of the location where most crimes are very ramped than the others.

References

<https://www.younglives.org.uk/>

<https://stackoverflow.com/questions/6077398/sql-query-using-a-case-statement-within-the-group-by-fields/>

<https://www.aspsnippets.com/Articles/Tip-Query-to-get-all-column-names-from-database-table-in-SQL-Server.aspx/>

[CREATE SCHEMA \(Transact-SQL\) - SQL Server | Microsoft Docs](#)

[Create Primary Keys in SQL Server - SQL Server | Microsoft Docs](#)

[SQL Server CAST\(\) Function \(w3schools.com\)](#)

[CREATE VIEW \(Transact-SQL\) - SQL Server | Microsoft Docs](#)

[How to create a view in SQL Server \(sqlshack.com\)](#)

[Create a Stored Procedure - SQL Server | Microsoft Docs](#)

[SQL Server create stored procedure \(15 ways\) - SQL Server Guides](#)

[SQL Server CREATE ROLE \(sqlservertutorial.net\)](#)

[Create a Server Role - SQL Server | Microsoft Docs](#)

[SQL Server Inner Join By Practical Examples \(sqlservertutorial.net\)](#)

[SQL SERVER JOINS Tutorial with Examples: INNER, LEFT, RIGHT, OUTER \(guru99.com\)](#)

https://blackboard.salford.ac.uk/ultra/courses/_132964_1/outline