



Master Thesis

submitted within the UNIGIS MSc programme
at the Centre for GeoInformatics (Z_GIS)
Salzburg University

Application and implementation of geocoding using Oracle Spatial component in Oracle11g

by

Nedim Dedić

UP40149

A thesis submitted in partial fulfilment of the requirements of
the degree of
Master of Science (Geographical Information Science & Systems) – MSc (GIS)

1st Thesis advisor: Dr. Adrijana Car
2nd Thesis advisor: Ao. Univ. Prof. Dr. Josef Strobl

Salzburg - Austria, 21. January 2010

Preface & Acknowledgements

I would like to thank to Dr. Adrijana Car for unselfish support, patience and guidance during my Master course at the UNIGIS Salzburg. Additionally, I would like to thank to all personal of Z_GIS (Centre for Geoinformatics Salzburg) at the University of Salzburg for wonderful cooperation, prompt responses and immediate actions when needed. This was wonderful journey through my education and I was more than honoured to work with this team.

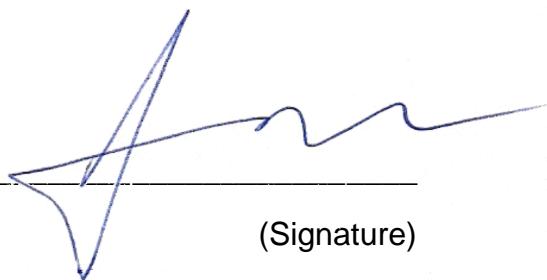
Special thank goes to my wife Zenina and my mother, who supported me during my studies at UNIGIS Salzburg and tolerated me in past years, lack of my free time and my “full day” commitment to studies. I know it was hard, but at the end – it was worth it!

Science Pledge

By my signature below, I certify that my thesis is entirely the result of my own work. I have cited all sources I have used in my thesis and I have always indicated their origin. The work has not been submitted previously for any other degree, nor is it under consideration by any other degree awarding body.

Salzburg, 21th January 2010

(Place, Date)

A handwritten signature in blue ink, consisting of a large, stylized initial 'A' followed by a series of connected loops and a long horizontal stroke extending to the right.

(Signature)

v | _____

Abstract

Importance of association of geographic coordinates with the real world addresses in everyday business in last couple of decades has become very obvious, especially for the purpose of determining and presentation of geographic locations of clients on interactive maps and spatial analysis of them. Companies cannot expect that clients, when purchasing products, provide them with geographical coordinates – they have to somehow associate them with a provided address.

It is important for the managers, database administrators and software developers to understand geocoding process and producing spatially referenced data from already collected addresses. This kind of data can be used in much wider spectrum than simple address, such as research, map visualisation and spatial analysis. The purpose of the work presented in this thesis is to investigate, evaluate and implement a process of associating geographic coordinates (geocoding) with available addresses using Oracle Spatial component in Oracle11g. This work is focused on and dedicated to Oracle geocoding functions that return only one matched value as a result. Additional purpose of this work is to find out what problems may arise during a geocoding process and with what quality a resulting set can be produced, with the focus on spatial accuracy.

The proposed approach of geocoding in this thesis allows efficient and simple transformation of available data to spatially referenced information without a need to acquire new data or additional software implementation. In addition, a new method for accuracy assessment of geocoding resulting set with Google Earth is proposed. The results of the study show that transformation is possible and executable, but very often spatially inaccurate data can be produced.

Contents

PREFACE & ACKNOWLEDGEMENTS	II
SCIENCE PLEDGE.....	IV
ABSTRACT.....	VI
CONTENTS.....	VIII
LIST OF FIGURES.....	XII
LIST OF TABLES	XIV
LIST OF LISTINGS	XVI
GLOSSARY	XVIII
1 INTRODUCTION.....	1
1.1 MOTIVATION.....	1
1.2 OBJECTIVES.....	1
1.3 PROBLEM DESCRIPTION	2
1.4 SCOPE.....	3
1.5 APPROACH.....	3
1.6 EXPECTED RESULTS	5
1.7 ISSUES THAT WILL NOT BE DISCUSSED HERE.....	7
1.8 INTENDED AUDIENCE	8
2 GEOCODING BASICS	11
3 DATABASE AND DATABASE MANAGEMENT SYSTEM AND THEIR APPLICATION IN GIS	13
3.1 INTRODUCTION TO DATABASE	13
3.1.1 Database model	14
3.1.2 Relational model.....	15

3.2	INTRODUCTION TO DATABASE MANAGEMENT SYSTEM	17
3.3	GEOGRAPHICAL DATABASE MANAGEMENT SYSTEM.....	19
3.4	STRUCTURED QUERY LANGUAGE INTRODUCTION	20
3.4.1	<i>Data Definition Language</i>	21
3.4.2	<i>Data Control Language</i>	22
3.4.3	<i>Data Manipulation Language</i>	22
3.5	PROCEDURAL LANGUAGE / STRUCTURED QUERY LANGUAGE	23
4	ORACLE ARCHITECTURE WITH FOCUS ON GEOCODING.....	25
4.1	ORACLE DATABASE ARCHITECTURE	25
4.1.1	<i>Physical Database Structure</i>	29
4.1.2	<i>Logical Database Structures</i>	30
5	ORACLE SPATIAL.....	31
5.1	DATA MODEL COMPONENT.....	33
5.2	LOCATION ENABLING COMPONENT	33
5.3	SPATIAL QUERY AND ANALYSIS COMPONENT	34
5.4	ADVANCED SPATIAL ENGINE COMPONENT	35
5.5	VISUALISATION COMPONENT	36
6	SDO_GEOMETRY DATA TYPE.....	39
6.1	SPATIAL GEOMETRIES IN ORACLE.....	39
6.1.1	<i>Points</i>	40
6.1.2	<i>Line String</i>	41
6.1.3	<i>Polygon and Surfaces</i>	42
6.1.4	<i>Solids</i>	42
6.1.5	<i>Collections</i>	43
6.2	ATTRIBUTES AND THEIR VALUES OF SDO_GEOMETRY DATA TYPE.....	43
6.2.1	<i>SDO_GTYPE Attribute</i>	44
6.2.2	<i>SDO_SRID Attribute</i>	46
6.2.3	<i>SDO_POINT Attribute</i>	47
6.2.4	<i>SDO_ELEM_INFO and SDO_ORDINATES</i>	47

7	GEOCODING WITH ORACLE SPATIAL	49
7.1	ORACLE GEOCODER	49
7.1.1	<i>Parsing</i>	49
7.1.2	<i>Address Search</i>	50
7.1.3	<i>Coordinates computation</i>	51
7.2	REFERENCE DATA	52
7.2.1	<i>Parameter tables</i>	52
7.2.2	<i>Data tables</i>	53
7.3	GEOCODER FUNCTIONS	54
7.3.1	<i>GEOCODE_AS_GEOMETRY</i>	56
7.3.2	<i>GEOCODE</i>	57
7.3.3	<i>GEOCODE_ADDR</i>	60
7.3.4	<i>GEOCODE_ALL and GEOCODE_ADDR_ALL</i>	61
8	IMPLEMENTATION OF GEOCODING AND ITS ACCURACY ASSESSMENT	
	63	
8.1	GEOCODING WITH GEOCODE_AS_GEOMETRY	63
8.1.1	<i>Implementation</i>	65
8.1.2	<i>Result</i>	68
8.1.3	<i>Analysis of the result</i>	71
8.2	GEOCODING WITH GEOCODE	77
8.2.1	<i>Implementation</i>	77
8.2.2	<i>Result</i>	78
8.2.3	<i>Analysis of the result</i>	82
8.3	GEOCODING WITH GEOCODE_ADDR	83
8.3.1	<i>Implementation</i>	84
8.3.2	<i>Result and Analysis</i>	84
9	SUMMARY	87
9.1	FINAL WORD	87
9.2	CONCLUSIONS	87
9.3	FUTURE WORK	89

REFERENCES.....	91
APPENDIX I.....	95
APPENDIX II.....	97
APPENDIX III.....	99
APPENDIX IV.....	101

List of figures

Figure 3-1: Diagram of an example database according to the Relational model.	16
Figure 3-2: A simplified database system environment (Rigaux, et al., 2002, Page 5)	17
Figure 3-3: Diagram showing user and database interaction using DBMS	18
Figure 4-1: The Client/Server Architecture and Distributed Processing (Source: Cyran, et al., 2005, Page 10-2); Part A: The client and server are located on different computers, and these computers are connected through a network; Part B: A single computer has more than one processor, and different processors separate the execution of the client application from Oracle.	27
Figure 4-2: A Multitier Architecture Environment Example (Source Ashdown and Kyte, 2009, Page 16-4)	28
Figure 4-3: Physical and Logical storage of Oracle (Source Ashdown and Kyte, 2009, Page 12-1).....	29
Figure 4-4: Segments, Extents, and Data Blocks within a Tablespace (Source Ashdown and Kyte, 2009, Page 12-2).....	30
Figure 5-1: Oracle Spatial technology components (Source Kothuri et al., 2007, Page 20).....	32
Figure 5-2: A sample map with multiple themes generated using MapViewer.	37
Figure 6-1: Example of spatial data that SDO_GEOMETRY can represent (Source Kothuri, et al., 2007, Page 56).....	40
Figure 6-2: Point representation of the cities surrounding Salzburg city (Source: Google Earth TM).....	41
Figure 6-3: Countries of Europe presented as polygons	42
Figure 6-4: Blank map of Croatia.....	43
Figure 6-5: Use of DESCRIBE SQL statement to retrieve information about SDO_GEOMETRY	44
Figure 6-6: SDO_GEOMETRY data type in Oracle.....	44
Figure 6-7: Example of the SDO_GTYPE in location column of schools_sf table .	46

Figure 6-8: Example of SRID values for schools_sf table	47
Figure 6-9: SDO_POINT_TYPE Data type.....	47
Figure 7-1: Oracle geocoder architecture Source: (Kothuri, et al., 2007, Page 153)	50
Figure 7-2: Interpolation example.....	51
Figure 8-1: Workflow diagram illustration the process of geocoding and analysis of the results.....	65
Figure 8-2: Preview of SELECT * FROM spatial.schools_sf table with OEM	67
Figure 8-3: Map presentation of San Francisco schools with MapViewer	70
Figure 8-4: Visual presentation of exported KML file in Google Earth	71
Figure 8-5: School position geocoded with GEOCODE_AS_GEOMETRY and presented as KML file with Google Earth and real school position (Map scale 1:95 meters)	72
Figure 8-6: Screenshot of distance measurement in Google Earth	74
Figure 8-7: Distribution of the buildings along Page St. (yellow line) in city of San Francisco and The Urban School of San Francisco	75
Figure 8-8: schools_sf table prepared for GEOCODE function	78
Figure 8-9: SELECT statement and results derived with GEOCODE_ADDR function.....	85

List of tables

Table 5-1: Example list of Advanced Spatial Engine components.....	35
Table 6-1: Values of D00T Format of the SDO_GTYPE attribute.....	45
Table 7-1: Parameter tables supported by Oracle geocoder.....	52
Table 7-2: Data tables supported by Oracle geocoder.....	53
Table 7-3: Comparing the Oracle geocoder functions regarding their input and output data structure	54
Table 7-4: Comparing the Oracle Geocode functions	55
Table 7-5: Match modes and their descriptions.....	58
Table 7-6: Match codes for geocoding operations and their description	59
Table 8-1: Deviations of the schools produced with GEOCODE_AS_GEOMETRY from their true location within Google Earth	75
Table 8-2: SDO_GEO_ADDR attributes and their values for GEOCODE function result	81

List of listings

Listing 3-1: Example of a DDL statement CREATE.....	21
Listing 3-2: Example of DCL statement GRANT.....	22
Listing 3-3: Example of DML statement SELECT.....	22
Listing 5-1: Location-enabling process for schools_of_san_francisco.....	34
Listing 5-2: Location enabling process for universities_of_san_francisco	34
Listing 5-3: Example of spatial query using SQL with location-enabled data	35
Listing 7-1: Syntax of GEOCODE_AS_GEOMETRY function.....	56
Listing 7-2: Example of street, city, state abbreviation and postal code on separate lines.....	57
Listing 7-3: Example of state abbreviation and postal code in one line	57
Listing 7-4: Example of city, state abbreviation and postal code in one line.....	57
Listing 7-5: Syntax of SDO_GEO_ADDR function.....	60
Listing 7-6: Example of calling the GEOCODE_ADDR function from dummy table DUAL.....	61
Listing 8-1: CREATE TABLE statement for creating San Francisco table in Oracle database	66
Listing 8-2: Geocoding in Oracle Spatial using GEOCODE_AS_GEOMETRY function.....	67
Listing 8-3: SELECT statement for id, name and location column over schools_sf table and returned result set.....	69
Listing 8-4: Updating location column of schools_sf table with GEOCODE function	78
Listing 8-5: SELECT statement and results derived with GEOCODE function	80
Listing 8-6: Updating location column using geocoder function GECODE_ADDR	84

Glossary

ACID - Atomicity, Consistency, Isolation, Durability

DBA – Database Administrator

DBID - Database Unique Identifier

DBMS – Database Management System

DCL – Data Control Language

DDL – Data Definition Language

DML – Data Manipulation Language

ERD – Entity Relationship Diagram

FOI – Feature of interest

GeoDBMS – Geographical Database Management System

GIS - Geographic Information System / Geographic Information Science

GML – Geography Markup Language

GPS - Global Positioning System

GUI – Graphical User Interface

HTML - HyperText Markup Language

HTTP - Hypertext Transfer Protocol

ISO - International Organization for Standardization

IT – Information Technology

KML – Keyhole Markup Language

MIT - Massachusetts Institute of Technology

OEM - Oracle Enterprise Manager

PL/SQL - Procedural Language/Structured Query Language

POI - Point of Interest

RDBMS - Relational Database Management System

RMAN - Oracle Recovery Manager

SDL - Software Development Laboratories

SDT – Spatial Data Type

SQL – Structured Query Language

SRID - Spatial Reference System Identifier

SVG - Scalable Vector Graphics

US NMAS – United States National Map Accuracy Standards

WGS - World Geodetic System

XML - Extensible Markup Language

1 Introduction

1.1 Motivation

In modern world and in modern business, possibility to adequately and effectively locate existing and potential customers on a map and analyse their spatial position has become a crucial tool for success for companies on the market. Possession of geographic coordinates derived from address data can help companies to solve problems regarding location of a new retail outlet, map consumer demand trends to best distribute products and advertising, scope digital advertising towards individual consumers and research consumer shopping patterns, and observe traffic within shopping centres and between retail outlets. It also helps in visualisation of market research findings and can help improving the overall planning ability of organisations. These data as well can be used to improve customer cooperation and involve them in certain activities.

There is a huge amount of companies and institutions that own their clients', customers' and partners' address information, but need to convert this data to geographical coordinates. Need for geographic coordinates and spatial reference of locations of e.g. customers is crucial for business within companies whose main focus is on product-to-home delivery, roads routing and optimization, locating of existing customers for service optimization, territorial planning, and site selection. The location is one of the key factors for such tasks.

1.2 Objectives

The main objectives of this thesis are to research, explain, implement and critically evaluate geocoding process with available data, using Oracle Spatial component of Oracle 11g and its geocoding functionality. This will include the following tasks and challenges:

1. Evaluation, analysis and explanation of geocoding in Oracle 11g and its functions
2. Analysis, research and explanation of databases and Oracle DBMS (Database Management System), their functionality, usability, availability and application for the purpose of geocoding
3. Analysis, evaluation and explanation of SQL (Structured Query Language) and PL/SQL (Procedural Language/Structured Query Language), and their possibilities and usability for the purpose of geocoding
4. Analysis of application of spatial data derived with Oracle geocoding functions for various purposes
5. Analysis of Oracle Spatial component and its application regarding spatial data management and geocoding
6. Empirical research of geocoder functionality in Oracle Spatial regarding quality and accuracy of geocoding resulting sets
7. Proposal for simple accuracy assessment of geocoded resulting set with Google Earth

1.3 Problem description

In everyday life, we use simple address data, as sort of information which helps us to locate specific point of interest (POI). Example of simple address is “*Town School for Boys, 2750 Jackson Street, San Francisco, CA*”. This address provides location information for mentioned school together with a name, but is not spatially referenced and does not hold any geographic coordinates. Collecting simple address data about clients is not a problem for most of the companies. During the process of customer purchase these data can be collected, mostly by already employed sales people, without a need for additional personal or additional tasks implementation. In some cases these have to be provided by customers to make contract. Examples are acquiring new credit card from a bank, making agreement

with insurance company and starting subscription for newspapers. This is not an exclusive list of the methods for acquiring information about clients. So, we can conclude that under certain circumstances acquiring simple address data about clients is a rather easy and simple job. In most cases it is easy to find personnel to do such tasks; usually there is already necessary personnel available.

However, a problem arises when these data need to be used for spatial analysis or for presentation on the map. So, to achieve this goal we have to execute a process which converts an address to geographical coordinates or, in other words, we need to *geocode* them.

1.4 Scope

Methodology used in this thesis defines and produces general methods and concepts of geocoding which can be used in various types of applications without scope limitation. The methods and concepts used and presented in this document can be adopted for further investigation and for different study areas, and as well for different amount of data. However, the study area used for this thesis is based on data from the United States, specifically for the city of San Francisco.

1.5 Approach

The work to be conducted here includes working with an Oracle database; thus my first step is theoretical investigation and description of databases. In addition, database management systems (DBMS) that are used as a “bridging tool” between database and user or client, and database models will be investigated and presented in general.

The language that is used as communication tool within databases, SQL, will be briefly described as well. Additional information, example listings and description of

PL/SQL (Oracle's specific procedural language) that will be used during geocoding implementation together with SQL will be provided to better understand the geocoding process. This part of the work will be entirely a theoretical study.

The following steps of the work to be conducted in this thesis will be based on additional theoretical investigation. This includes consulting of literature, user manuals and available case studies. Thus, derived results of this investigation regarding Oracle 11g with special focus on its physical and logical structure will be presented.

As Oracle Spatial component is a crucial tool for the work to be conducted here, its geocoder implementation and geocoding functionality will be examined and presented. This will be followed by investigation and description of a location enabling process and spatial query. Analysis of Oracle Spatial data model, architecture and functionality will be presented as well.

SDO_GEOMETRY data type in Oracle Spatial is used for manipulation and storage of spatial data, which are needed to demonstrate functionality of geocoder. Thus, this data type will be briefly introduced as well. Description of type, attributes and values of SDO_GEOMETRY data type will be provided.

Geocoding, geocoding engine, and Oracle geocoder and its functions (GEOCODE_AS_GEOMETRY, GEOCODE, GEOCODE_ADDR), which are used in the process of geocoding implementation, will be examined. In addition a brief introduction of GEOCODE_ALL and GEOCODE_ADDR_ALL will be given as a base for further research of geocoding process, which return more than one matched result. The main aim here is to gain new and concise insight to the above described topics that can be used as a good foundation for the purpose of implementation of geocoding process.

After research and description of all crucial and important aspects for geocoding process within Oracle Spatial, the implementation part of the geocoding project will be done and this part of the work will be based on empirical methods, such as Action Research. In Action Research, the researchers attempt to solve a real-world problem while simultaneously studying the experience of solving the problem (Davison, Martinsons and Kock 2004). In this thesis, we work on a real-world problem of geocoding with Oracle Spatial, while simultaneously studying the experience during the this process. Geocoding functions in Oracle Spatial tool, which return only one match as a result (GEOCODE_AS_GEOMETRY, GEOCODE, GEOCODE_ADDR), will be used on existing sample data to produce a resulting set of geocoded addresses.

This set will then be examined regarding accuracy using visual representation, such as presenting results on the map. First, data produced from geocoding process will be exported as Key-hole Markup Language (KML) file and then, in order to visually compare to already existing maps provided from Google, imported into Google Earth. It is important to say that, place and position and road detail vary greatly from place to place in Google Earth. They are most accurate in North America, Europe and Australia (Wikipedia 2011). Thus, we can use Google Earth as accuracy validation tool, but with a dose of caution. Large map scale will be used for visual representation of the geocoding results. Additional SQL and PL/SQL sample listings for the purpose of geocoding of existing data will be produced and presented.

1.6 Expected results

Most of the literature available today, such as Kothuri, et al. (2007), refers to the methods that work with “non saved data” to explain or evaluate geocoding process. Term “non saved data” is used to explain data that are not actually saved in database. Example of using “non saved data” in geocoding process is when we hard code address data in geocoding functions. The main goal of this master thesis

is to explain new method of geocoding of existing address data (already saved in database) in Oracle with Oracle Spatial geocoding component, and assessment and validation of the accuracy of geocoded resulting set. Usability and application of these data will be evaluated regarding situations where good horizontal positional accuracy is needed. Positional accuracy is an assessment of the closeness of the location of spatial objects in the dataset in relation to their true positions on the earth's surface (PSMA Australia May 2010).

Spatial coverage of our sample data set is the City of San Francisco, therefore U.S. National Map Accuracy Standards (US NMAS) will be used as a base for accuracy assessment of produced resulting set from sample data. Possibility of presentation of derived results on the map with Oracle MapViewer and external tools such as Google Earth and its accuracy will be explored and evaluated.

Even so, this thesis is mostly based on the previous research of Kothuri, et al. (2007), it cannot be consider as work duplication. In addition to sound fundamentals, this document aims to producing new insight and new knowledge about the process of geocoding, especially when dealing with already collected and saved data. A method for a simple accuracy assessment of geocoding results with Google Earth is provided in addition. In this thesis the following questions are expected to be answered:

- 1) Is it possible to convert addresses to geographical coordinates with good positional accuracy, according to US NMAS specification, using Oracle Spatial geocoder?
- 2) How to convert available addresses to geographical coordinates using Oracle Spatial?
- 3) Is it possible to perform accuracy assessment of geocoded results with Google Earth as a reference tool for accuracy validation?

- 4) Can geocoded data be presented on the maps and integrated into web mapping services as Google Maps, Yahoo Maps, etc?
- 5) What can we expect from geocoded data in the terms of validity of resulting set, except geographical coordinates? In other words, how valid are other data beside geographic coordinated, such as street name, city name, etc?

1.7 Issues that will not be discussed here

Even through foundation of this work is theoretical study and empirical evaluation of the geocoding process with Oracle geocoder and its functions, methods researched, presented and evaluated here are based only on geocoder functions that return one match for geocoded address. Limitation on these functions in work presented here is mostly because lack of time and because working with other functions requires additional effort that can be considered beyond one master thesis. Thus, functions that return more than one match are not in the focus of this thesis. However, some insight in their architecture and application will be presented as a base for further research.

Working with non-standardized, incomplete or ambiguous input data and their manipulation for the purpose of usability with geocoding functionality will not be discussed here as well. SQL and PL/SQL and their core functionality which are required to obtain results, will be presented in thesis, while other explanation and evaluation of these languages will not be presented.

Some additional tools that are required to present results or execute commands (examples: Oracle MapViewer, SQLPlus, etc) will be used here. This thesis however does not cover their evaluation or description.

1.8 Intended audience

This thesis is expected to be of interest to:

- 1) People interested in challenges of transforming already existing addresses to geographical coordinates
- 2) Database administrators with special focus on spatial data and their presentation
- 3) Logistics, advertising and publishing companies, whose main focus of business lays in delivery
- 4) People working with Oracle Spatial
- 5) Software developers and mapping experts with interest in mapping and analysis of customer or client data
- 6) People interested in research regarding geocoding in general

1.9. Thesis structure

This document is divided in several chapters as follows:

Geocoding basics	Short introduction to geocoding in general.
Database and Database Management System with their application to GIS	This chapter provides an introduction to database, database models relevant to geocoding, and database management systems and their application, usability and functionality. Basic concepts of GeoDBMS will be presented as well. This chapter covers SQL and PL/SQL, their functionalities, while additional generic samples of code will be provided.

Oracle architecture with focus on geocoding	Introduction and evaluation of Oracle database and Oracle Database Management System, architecture, logical and physical structure with insight to relevance to geocoding.
Oracle Spatial component	Evaluation and analysis of Oracle Spatial components, its architecture, availability, usability and functionality with spatial data.
SDO_GEOMETRY data type	Analysis and evaluation of SDO_GEOMETRY data type and its functionality regarding storing and modelling different types of location information.
Geocoding with Oracle Spatial	Explanation and implementation of the process of geocoding in Oracle 11g. Functions that return only one matched value as a result will be analysed and presented, while additional information about functions that return more than one values as result will be briefly described as a base for further research. Example listing of this process and will be presented.
Implementation of geocoding and accuracy assessment	Description of implementation of geocoding based on concepts and methods presented in thesis. In addition, method for accuracy assessment of geocoded result with Google Earth is presented as well.
Summary and Future work	Summarizing work presented in thesis and deriving

conclusions from them. Further research steps will be presented as well.

References

List of references

Appendix

Additional listings, figures and tables

2 Geocoding basics

“Geocoding is the process of finding associated geographic coordinates from other geographic data, such as street addresses, or zip codes.” - (Miller, et al., 2010)

One of the biggest advantages of GIS over other similar technologies, when dealing with space, is its capability to integrate data from different sources. But to work with this data simultaneously they have to be spatially referenced. In other words, all attribute data must be associated or in some way referenced to a spatial unit, allowing spatial analysis and mapping. This procedure of association is defined as geocoding (Brinker and Minnick, 1995).

Thus, geocoding is the process of finding associated geographic coordinates (latitude or longitude) from other data provided in the form of street addresses, postal codes, etc. It can be performed either manually or using a computer (Brinker and Minnick, 1995). According to definition in geocoding case study provided by Melnick (2002), when speaking about computer based geocoding, we say that is the process by which the GIS software matches each record in attribute database with the geographic file. Beside the purpose to associate geographical coordinates with address, geocoding serves as well as a tool for address normalization which involves structuring and cleaning an input address.

Geocoded locations are specifically useful in GIS analysis and cartography tasks. Some scientists consider making maps, or cartographic presentation as a primary purpose of GIS (Melnick 2002). Geocoding is very much applied to the internet and used by services that offer possibility like finding and routing driving directions from location A to location B, or to find nearest restaurant or store that is geographically nearest to a user. It is one of the several methods for obtaining coordinates for getotagging media.

Very simple method, used by geocoding is address interpolation. Interpolation is a method of constructing new data points within the range of a discrete set of known data points. In case of geocoding, interpolation process obtains new geographic coordinates data for certain location from already defined geographical coordinates. We use it to estimate a value of coordinates at an undetermined location, like address or street name with number, from measurements of already known locations.

“This method makes use of data from a street geographic information system where the street network is already mapped within the geographic coordinate space. Each street segment is attributed with address ranges. Geocoding takes an address, matches it to a street and specific segment. Geocoding then interpolates the position of the address, within the range along the segment” - (Wikipedia - Free Encyclopedia, 2010).

The same method of geocoding in Oracle Spatial is used in this master thesis that creates coordinates for school locations, which are unknown, from already know locations from referenced data. Because this chapter has an aim to only introduce geocoding in general, we explain geocoding process in Oracle Spatial, together with its functionalities in chapters six and seven.

To perform process of geocoding and to conduct it in a proper manner, understanding of underlying processes and software architecture is crucial. In the following chapters, we study, describe and evaluate various tools and software that are required to perform process of geocoding in Oracle, before we focus on implementation of geocoding.

3 Database and Database Management System and their application in GIS

This chapter provides introduction to the basics of database and Database Management System (DBMS) and gives an overview of common database model and systems for database management which are used for the purpose of geocoding. Work conducted here is based on computer geocoding with focus on geocoding existing address data in a database. Thus, it is important to understand the concept and functionalities of databases and Database Management System (DBMS) when working with geocoding, especially when working with geocoding in Oracle.

3.1 Introduction to database

As we all probably know, databases became integral part of everyday life in modern society and all the time during our “modern life” we come across situation where interaction with database is necessary to fulfil our intentions or to finish our requirements. Most common situation of interaction with database is checking our account status in bank, making reservation for hotels, airplane flights, paying bills at cash register, and so on. This list can be pretty large.

A database is a collection of related data stored in standardized format, capable of being shared by multiple users (ISRD-Group 2006). Krishna (1992) defines database as collection of the data and it can cover data in all forms, in addition to data stored on computer, is as well acceptable from the perspective of database definition.

A simple example of a database is a phone book or library. In both cases data are organized and can be easily accessed, because they are all connected and indexed in specific manner suitable for intended purpose. Another example of a

database, when speaking about storing data for geocoding purpose, is a collection of geographical information such as street name, house number, postal code, state and country, which are interconnected in specific way each with another. These data, if organized properly, can be used in a process of geocoding.

In geoinformatics, database can be defined as organized collection of spatial data for multipurpose and saved in digital form. In this case data must be organized and classified in a structural format and by description in metadata. Structural format is definition which means that data must follow rules of storage and interaction for certain database. Metadata describe the actual data. Together, metadata and data form one logical entity of methods for data organization and enable effective access to them.

3.1.1 Database model

The word “model” in information technology stands for abstract description of reality. A database model is described in a formal language and supported by the DBMS. In some cases database model is defined as database schema, while structure of database is defined as format of database. There are several ways of using the term “database model” when dealing with computer science. It is very important to differentiate them. As defined by Rob and Coronel (2009) a data model can be considered as relatively simple representation of more complex real world data structures. According to Rob and Coronel (2009) definition data model, within database environment, represents data structures and their characteristics, constraints, transformations, relations, and other constructs with purpose of supporting a specific problem domain.

When speaking about database in digital form, there are four common database models which are most often in use:

- 1) Hierarchical model

- 2) Network model
- 3) Relational model
- 4) Object-relational model

Oracle database, which is used in this work as one of the tools, uses a relational model. Sample data we use for the purpose of geocoding in implementation part of this master thesis are saved in relational database. Thus, in this part of the work, only this model is explained.

3.1.2 Relational model

The relational model for database management is a database model based on first-order predicate logic, first formulated and proposed by Codd (1969). The idea is that a database consists of a series of unordered tables which can be manipulated using non-procedural language (par example: SQL) that return tables. This model had completely different approach of database implementation then traditional. During the last twenty years, it is shown that this model is quite more flexible, less complicated and more functional then other models defined before this one.

To better understand the relational model, Codd (1990) has developed a definition which states that a table in SQL database schema corresponds to a predicate variable; the contents of a table to a relation; key constraints and SQL queries correspond to predicates. Data items are organized as a set of formally described tables (Figure 3-1). Database that uses relational model is well suited for “ad hoc” user queries. In database terminology “ad hoc” term represents reporting system that allows the users themselves to create specific, customized queries. Geocoding, for example, and spatial querying are these kinds of queries. Thus, we say that “ad hoc” queries are very important aspect of GIS analysis (Kutz 2004).

In Figure 3-1 we see relational model that has three tables each of them representing school names, cities and zip codes. Using this model and with the help of additional software (example is SQL) we can access any data in many ways without having to reorganize the database tables. This allows us to create and use relations within data from various tables. In geocoding process, for example, common fields from tables (Figure 3-1), such as “city” and “zip code” are used to “attach” coordinate information to the “school name” table and store them in empty “location” column. Relational model makes possible to select from which table to use, for example, zip codes column and how to relate it within city column during the process of geocoding. I will explain this in detail in implementation part of this work.

Relational Model

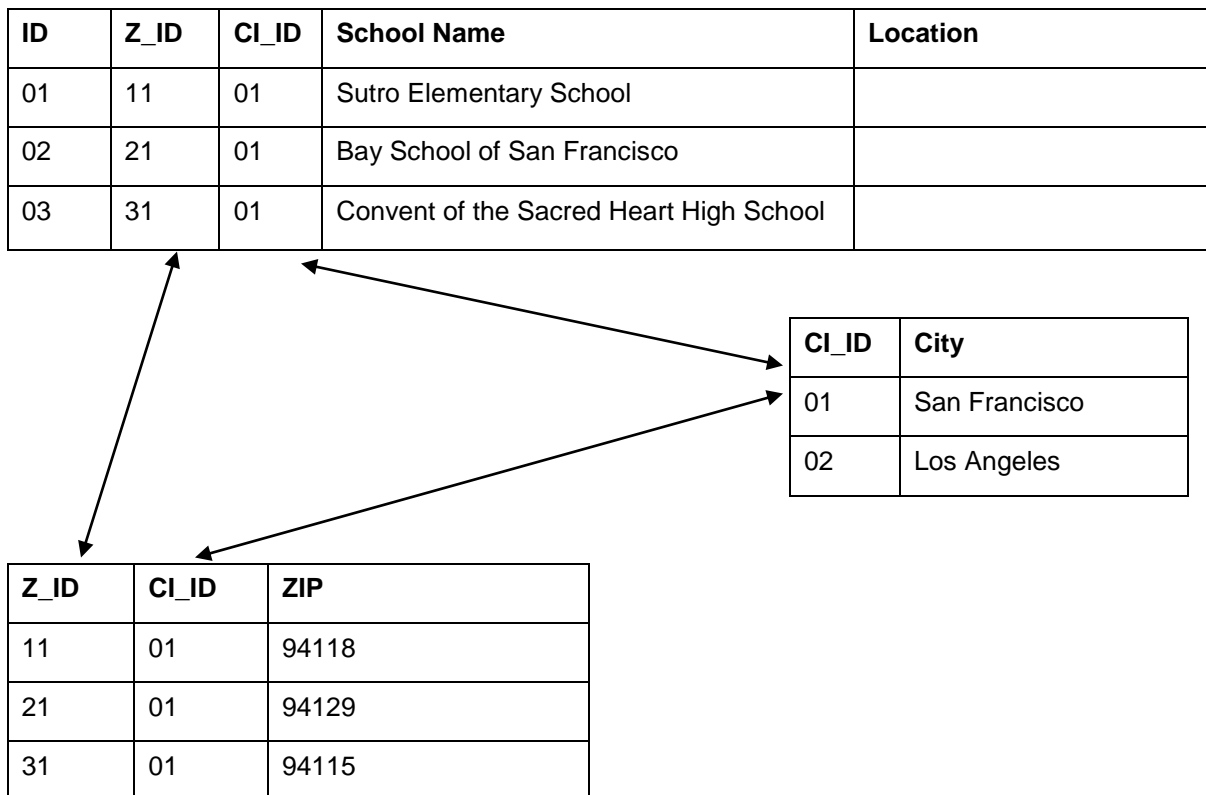


Figure 3-1: Diagram of an example database according to the Relational model.

3.2 Introduction to Database Management System

A Database Management System (DBMS) is a collection of software or more common as computer programs which give possibility of controlling creation, maintenance, and the use of a database. DBMS allows entities, such as companies and organizations to place control of database development in the hands of database administrators (DBA) and other specialists. This is a very important tool in the process of computer geocoding. It allows us to use geocoding functionalities in much easier and understandable environment. In the work presented in this master thesis we use Oracle DMBS and applicable tools to perform geocoding. Figure 3-2 represent simplified database system environment.

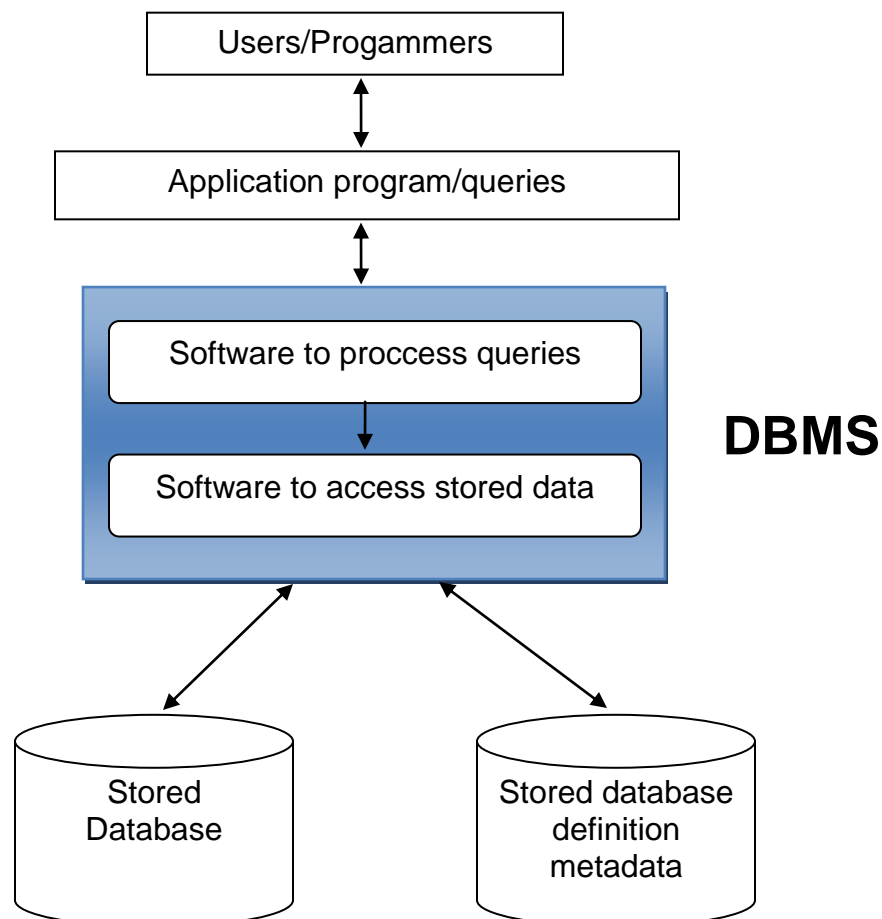


Figure 3-2: A simplified database system environment (Rigaux, et al., 2002, Page 5)

The biggest advantage of a DBMS is a possibility that allows different user application programs to access the same database. DBMS may use any of a variety of database models, such as the network model or relational model explained in section above. Also, DBMS allows users and other software to retrieve and store available data in a structured and organized manner. Instead of implementing additional software, user can provide questions directly to a database using a query language. DBMS provides the ability to logically present database information to users and applications.

We have a variety of DBMS definitions, but very good and concise definition is provided by Sumathi and Esakkirajan (2007), where they state that database management system is a collection of interrelated data and set of programs to access that data. Simple and concise definition of DBMS as well is provided by (Narang 2006), where he states that DBMS is a software package that controls all access to database.

Simple presentation of user interaction with database through DBMS is shown in Figure 3-3: User uses DBMS to manipulate database data and database answers to user through DBMS.

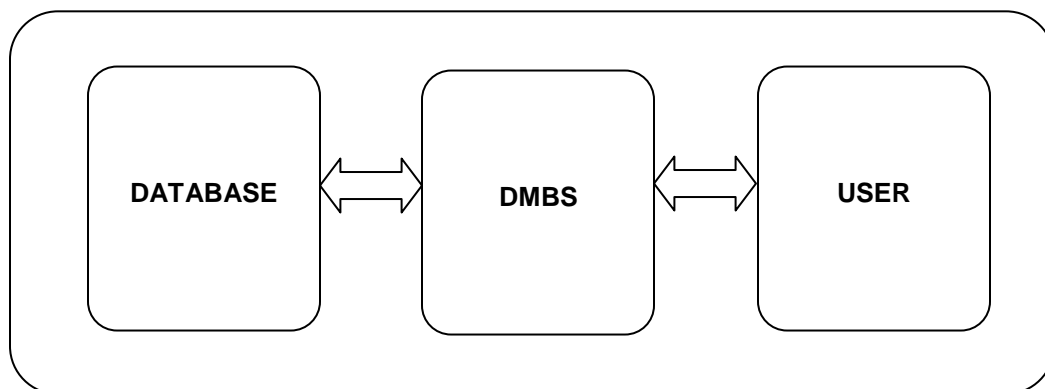


Figure 3-3: Diagram showing user and database interaction using DBMS

3.3 Geographical Database Management System

Geographical Database Management System (GeoDBMS) can be seen as a piece of software or additional software module that has capability of working with underlying DBMS. In addition to normal DBMS it has a possibility to support spatial data, spatial data types (SDT) and query language from which these SDTs are manipulative. Spatial indexes, spatial information processing and domain specific rules for query optimization are as well supported by GeoDBMS.

If we take a look at collection of data, par example telephone book , which is stored in some form of digital database and manipulated by a certain DBMS, we would probably find data like people names, address data like street, city, postal code and country. This data can be manipulated using DBMS and query language, so we can query people by street, by zip or any other attributes as well. What we cannot do is spatial analysis or manipulation with spatially related data. To achieve this, we must have a DBMS and database that supports spatial data and working with them, and off course, we must have spatial data (in this case geographical coordinates). If all requirements are satisfied, we will also have possibility to do spatial queries, spatial analysis, like examining what amount of people live in buffer zone of several kilometres for certain zip, we will have possibility to present locations on the map and to execute spatial queries and analysis.

GeoDBMS is crucial in Geoinformation Science & Systems and can be used by applications in Astronomy, Multimedia Information Systems, Logistics, etc. GeoDBMS enables enterprise computing based on spatial data. GeoDBMS is a part or an extension of DBMS, with additional requirements which needs to be fulfilled. Rigaux et al. (2002) defined methods and objectives that certain extension of DBMS need to fulfil to be considered GeoDBMS:

- 1) Logical data representation must be extended to geometric data, while satisfying data independence principle and keeping its simplicity

- 2) New functions of query language need to be integrated to satisfy requirements of operations applicable to geometric objects
- 3) Physical representation of spatial data should be present
- 4) Additional algorithms in the field of relational query processing are needed
- 5) Efficient data access including spatial data must be provided

For the geocoding of data in a database, it is a crucial requirement that database has possibility of storing and manipulation of spatial information. The probably most known DBMS which supports manipulation with spatial data is Oracle with Spatial component. We discuss and focus our study on Oracle and Oracle Spatial component in following chapters.

3.4 Structured Query Language Introduction

SQL stands for “Structured Query Language”. It is a relational database language and is used for communication with DBMS (Throll and Bartosch, 2010). Commands in SQL are “English-like” statements and are used for query, insert, update and deletion of data, schema creation and modification, and data access control. It is a non-procedural database language, which means that we tell SQL what data to retrieve rather than how. SQL is very important for the implementation of the process of geocoding. In Oracle, for example, we use SQL together with Procedural Language/Structured Query Language (PL/SQL) to geocode geographical information and produce geographical coordinates. SQL gives possibility to use all necessary functions to retrieve information required in the process of geocoding, while PL/SQL gives possibility to geocode them. Because, we use SQL in the geocoding process, short introduction to this language is presented in this work.

SQL in general is considered as a sublanguage, because of its non-procedural nature. Nevertheless it is a programming language which can be used for

maintenance of database objects and manipulation of the data within the object. It is a universal language for data access and it became a standard language for database management. Almost every system for database controlling uses certain model of SQL language to manage a database. Most know databases which use SQL are: Oracle, MySQL, DB2, SQL Server and PostgreSQL.

According to Oppel and Sheldon (2008) it is possible to categorize SQL statements according to the functions they perform. Thus, we can separate SQL statements into three categories: Data Definition Language (DDL), Data Control language (DCL) and Data Manipulation Language (DML).

3.4.1 Data Definition Language

These statements are used to create, delete or modify database objects (tables, views, triggers, etc). Most common DDL statements are **CREATE**, **ALTER** and **DROP**. An example of **CREATE** statement is shown in Listing 3-1. In this example, DDL statement is used to create new table in a database with the name `schools_sf` and with table fields `id`, `name`, `addr_number`, `street`, `postal_code`, `city`, `state_abrv` and `location`.

```
CREATE TABLE schools_sf (  
  id NUMBER,  
  name VARCHAR2(100),  
  addr_number VARCHAR2(10),  
  street VARCHAR2(50),  
  postal_code VARCHAR2(5),  
  city VARCHAR2(20),  
  state_abrv VARCHAR(2),  
  location SDO_GEOMETRY  
);
```

Listing 3-1: Example of a DDL statement CREATE

3.4.2 Data Control Language

This category of statements allows us to control who or what kind of application has access to specific objects in a database. With two primary DCL commands, **GRANT** and **REVOKE**, we can grant or restrict access to certain objects in database to certain users or roles and type of access. In previous example, Listing 3-1, we showed example of table creation with name `schools_sf`. Under assumption that we already have user `nedim` in our database, we can grant all privilege to user `nedim` over table `schools_sf` with code provided in Listing 3-2.

```
GRANT ALL ON schools_sf TO nedim;
```

Listing 3-2: Example of DCL statement GRANT

3.4.3 Data Manipulation Language

DML statements are used to retrieve, add, modify, or even delete data stored in database objects (Oppel and Sheldon, 2008). **SELECT**, **UPDATE**, **DELETE** and **INSERT** are statements that are most used when working with SQL and these are primary statements associated with DML. In the example provided in Listing 3-3. we use **SELECT** statement to retrieve all data from `schools_sf` table where `name` is equal `Sutro Elementary School`.

```
SELECT * FROM schools_sf  
WHERE FirstName = 'Sutro Elementary School';
```

Listing 3-3: Example of DML statement SELECT

3.5 Procedural Language / Structured Query Language

PL/SQL is a crucial tool for geocoding in Oracle. All geocoder procedures within Oracle database are written in this language. Without PL/SQL, geocoding in Oracle would be impossible. Every procedure we use in this work is constructed from PL/SQL code. Thus, it is important to understand fundamentals about this language.

PL/SQL is Oracle Corporation's procedural extension language for SQL and the Oracle relational database. Computer language is a particular way of giving instructions to a computer, while “procedural” refers to a series of ordered steps that one computer needs to execute and present or produce a set of results. The speciality of this type of language is that it includes data structures, which hold information that can be used multiple times (Pribyl and Feuerstein, 2002).

Procedural language allows a programmer or developer to define an order of steps to be followed in order to produce a result. For example, with PL/SQL we define steps that need to be followed during the process of geocoding. We can consider PL/SQL as language that is closely related to SQL, but in addition, this language allows us to write programs as an ordered series of statements. Much of the syntax of PL/SQL is borrowed from Ada, and it support named program units and packages (Pribyl and Feuerstein, 2002).

PL/SQL is developed as addition to SQL within Oracle database, because SQL is not general purpose language, which can express computer algorithms. Even so, there is a possibility to write SQL code that can consist of the sequences of SQL statements, it has no possibility to have conditional statements. In simple words, SQL has no possibility to say something like: “If something is TRUE then execute something or return particular part of result set, OTHERWISE do something else”.

PL/SQL has possibility to implement such commands and to execute it within Oracle database.

The advantages of PL/SQL over other languages when working with Oracle database are: tight integration with SQL, better performance, higher productivity, full portability, tight security, access to predefined packages, support for object-oriented programming and support for developing web applications (Moore, 2007). All of these advantages are huge plus for PL/SQL when speaking about geocoding. Here, I will briefly explain some of them:

- 1) Tight integration with SQL is important because we are able to use SQL queries without additional software engineering in PL/SQL. Most of the other languages require writing of additional code to integrate SQL queries – PL/SQL does not.
- 2) Performance plays a huge role when working with geocoding. Most of the time, geocoding is the process that requires transformation of thousands or millions of data. Thus, only small amount of better performance during geocoding of one address can be extremely useful with large amount of data.
- 3) Security of the data, especially with large companies and governmental institutions is always on the top of the list of priorities, for example, geocoding of military data for the purpose of marking the movement of infantry.

4 Oracle architecture with focus on Geocoding

The Oracle Database, also known as Oracle DBMS, or simply as Oracle, is relational database management system, which is designed, produced and marketed by Oracle Corporation. In last twenty years Oracle became a leading brand in the field of systems for database management.

Larry Ellison and his co-workers, Bob Miner and Ed Oates founded consultancy Software Development Laboratories (SDL) in 1977. SDL developed the first and original version of the Oracle software. The name Oracle comes from the code-name of a CIA-funded project Ellison had worked on while previously employed by Ampex (Schofield and Brockes, 2010).

Oracle is a database and DBMS that is used in the process of geocoding in this master thesis. Thus, knowledge of the architecture of this database with application to geocoding is useful to better understand the process itself.

4.1 Oracle Database Architecture

As it can be derived from previous chapters, a database is a collection of data treated as a unit. Sorting and retrieving related information is the purpose of every database. As a “key” to information management, database server manages data and multiuser environment, so that users can concurrently access the same data (Strohm, 2008). Database server as well has a purpose to stop unauthorised access to data, provide efficient solutions and in case of need, failure recovery.

“Oracle Database is the first database designed for enterprise grid computing, the most flexible and cost-effective way to manage information and applications.

Enterprise grid computing creates large pools of industry-standard, modular storage and servers. With this architecture, each new system can be rapidly provisioned from the pool of components. There is no need to provide extra hardware to support peak workloads, because capacity can be easily added or reallocated from the resource pools as needed.” – (Strohman, 2008, Page 1-1).

As mentioned above, Oracle database is designed for enterprise grid computing, which can be considered as huge advantage in the field of Information Technology (IT). Grid computing produced resilient and lower cost enterprise systems (Joseph and Fellenstein, 2004). Grid computing in other words can be seen as distributed systems, but with non-interactive workloads that involves large number of files. The grid computing allows groups of independent, modular hardware and software components to be connected and rejoined on demand to meet the changing needs of business (Strohman 2008).

Oracle Application Architecture is constructed by two most common database architectures: a) client/server and b) multitier. Client/server architecture enables the roles and responsibilities of a computing system to be distributed among several independent computers. These computers are known to each other only through network. Additional advantage for this architecture is also greater ease of maintenance; all data are stored on the servers with better security controls, and through them better access and resource control can be achieved. In Figure 4-1 simple client/server architecture is presented. As we can see, clients can access to database server through network. All data are stored on database server.

To better understand client/server architecture for the purpose of geocoding, we can present client from Figure 4-1 as a user or programmer who needs to geocode information stored on database server. Because, there are lot of information that need to be geocoded, we have stored them on several database servers. Thus, user or programmer initiates the process of geocoding on the client machine, which then, through commands sent via network, starts execution on database server. If

there is a huge amount of information, more database servers are used. They are as well connected and communicate via network.

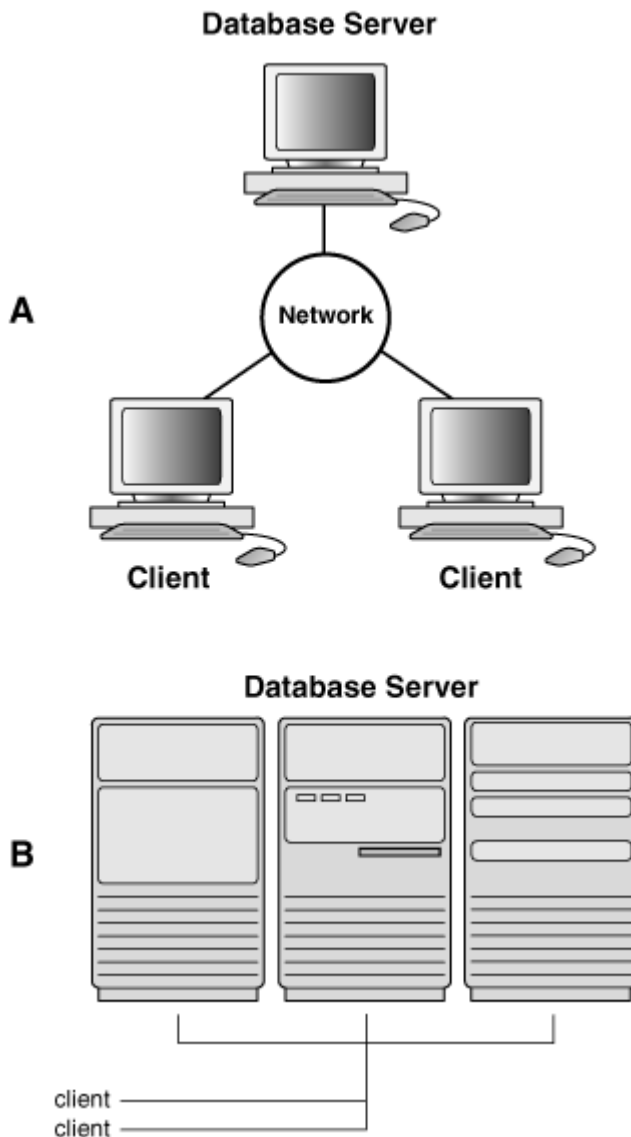


Figure 4-1: The Client/Server Architecture and Distributed Processing (Source: Cyran, et al., 2005, Page 10-2); **Part A**: The client and server are located on different computers, and these computers are connected through a network; **Part B**: A single computer has more than one processor, and different processors separate the execution of the client application from Oracle.

Multitier architecture is also a sort of client-server architecture, but here, the application processing, and the data management are logically separated

processes. As we can see in Figure 4-2, several clients access to Application Server 1, which then contacts one of database server, or additional application servers and execute or delegate certain job. In any case, this system prevents certain servers to overload and to perform better, because they do not serve all necessary applications or store all required databases. It is much easier to maintain and secure system organized like in Figure 4-2 than in Figure 4-1 (Joseph and Fellenstein 2004). This is a very useful and recommended structure when dealing with large amount of spatial information. Also, this structure is recommended when geocoding large amount of information. Process of geocoding is similar as with client/server architecture, except, we have application servers in addition here which allows better functionality and more speed.

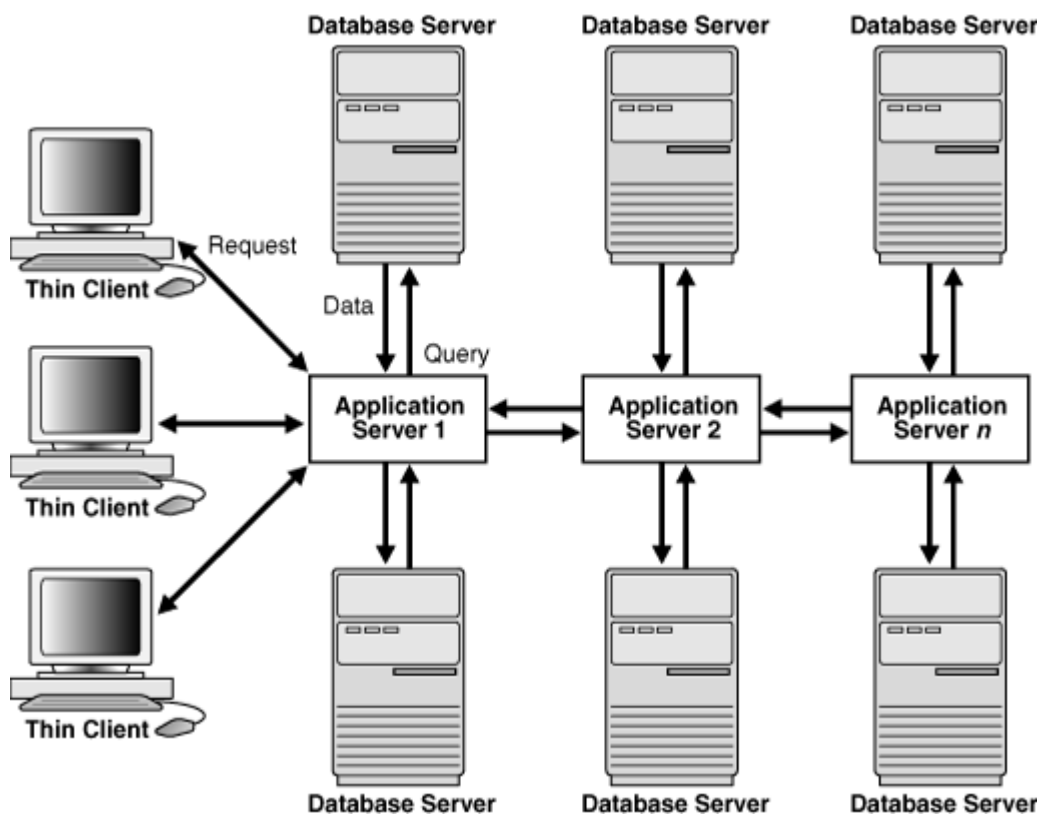


Figure 4-2: A Multitier Architecture Environment Example (Source Ashdown and Kyte, 2009, Page 16-4)

As most of other databases, Oracle as well, has logical and physical structures. In case of Oracle, physical structures can be manipulated without affecting logical storage structures, because physical and logical structures are separated. In Figure 4-3 an entity-relationship diagram for physical and logical storage of Oracle is presented.

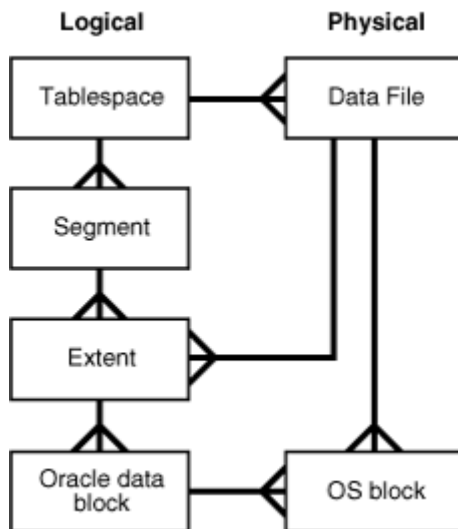


Figure 4-3: Physical and Logical storage of Oracle (Source Ashdown and Kyte, 2009, Page 12-1).

4.1.1 Physical Database Structure

Characteristic of every RDBMS is independence of logical data structures from physical database structure. Because they are separated we have possibility to manage and influence physical storage without affecting access to logical structures. The very basic example is renaming a database file. If we rename a database file in Oracle it would not change or rename tables stored inside Oracle database. In the case of geocoding, physical database structure is represented as “real files” on database server that holds information we need for geocoding and geocoded data as well.

When speaking about physical structure of Oracle 11g, we mean the physical attributes of database, like type of files in the database (Kogent Solutions Inc., 2008). The major components of physical structure are: Data Files, Redo Log Files and Control Files.

4.1.2 Logical Database Structures

Beside physical files used by database, Oracle maintains many types of logical structures. These structures enable Oracle Database to “fine-tune” control over disk space usage. “The logical units of database space allocation are data blocks, extents, segments, and tablespaces.” - (Ashdown and Kyte, 2009, Page 12-1). Figure 4-4 explains the relationships between data blocks, extents, and segments within a tablespace.

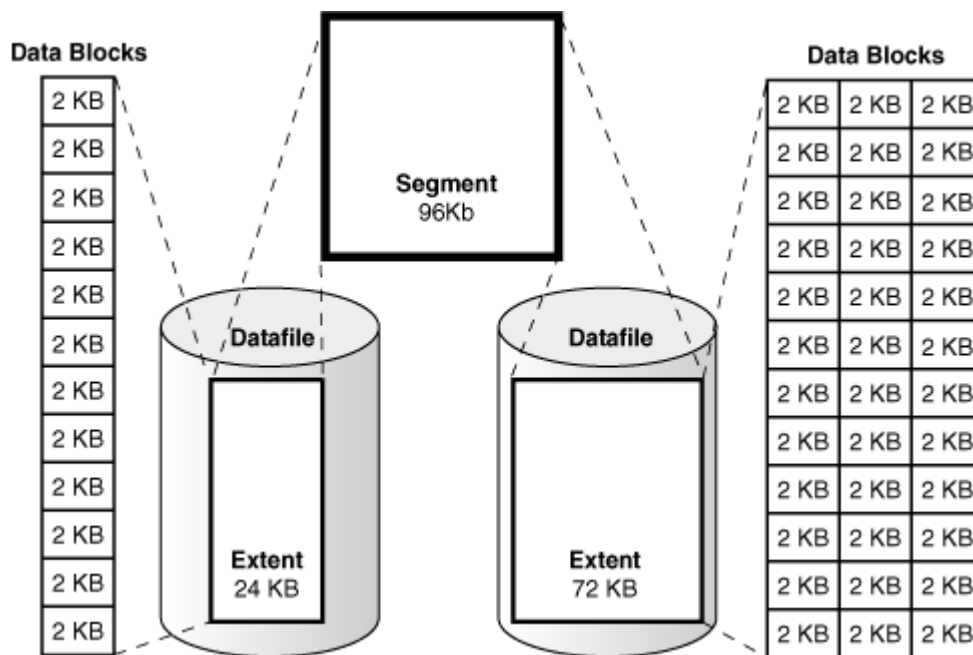


Figure 4-4: Segments, Extents, and Data Blocks within a Tablespace (Source Ashdown and Kyte, 2009, Page 12-2)

5 Oracle Spatial

This chapter has an aim to explain Oracle Spatial component of Oracle 11g. Oracle Spatial is a separately licensed option component of the Oracle Database. Its main purpose is to help users to manage geographic and location data in a native type in Oracle Database. It supports a wide range of applications such as automated mapping, geographic information systems (GIS), wireless location services, location-enabled e-business, etc. However, in the last two editions, subset of Oracle Spatial functionalities is provided at no cost policy under name Oracle Locator (Ashdown and Kyte, 2009).

It is very important to say that Oracle Spatial complies with the implementation directive of the OpenGIS Consortium – Simple Features Guidelines. Government agencies land management, infrastructure, energy explorations and distribution, and data warehousing are targeted applications for Oracle Spatial (Rigaux, Scholl und Voisard 2002).

Besides advantages discussed above, Oracle Spatial has accessing and language features that are keyed to the web and internet. It uses open programming standards of SQL, Extensible Markup Language (XML), Hypertext Transfer Protocol (HTTP), HyperText Markup Language (HTML), Java, .NET, Geography Markup Language (GML), OpenLS, ISO211 and Scalable Vector Graphics (SVG) (Pick 2008). Huge numbers of standard types of spatial functions are available within Oracle Spatial, like SQL spatial type, R-tree index that can use coordinates, various spatial operators and functions, and geodetic function that supports use of latitude/longitude coordinate system. Additional standard functions can handle indexing, querying, loading, linear referencing, partitioning and even aggregation of spatial data.

Relationship determination via spatial analysis between layers is also one of possibilities of this component. Higher level spatial functions include GeoRaster Type, Network Data Model, Topology Data Model, Geocoding Engine, RoutineEngine, Spatial Data Mining, Text Annotation Type, GML/SVG support and Transportable Tablespaces.

Oracle Spatial technology components can be separated into five categories (Figure 5-1): Visualization, Location Enabling, Data Model, Query and Analysis and Advanced Spatial Engine.

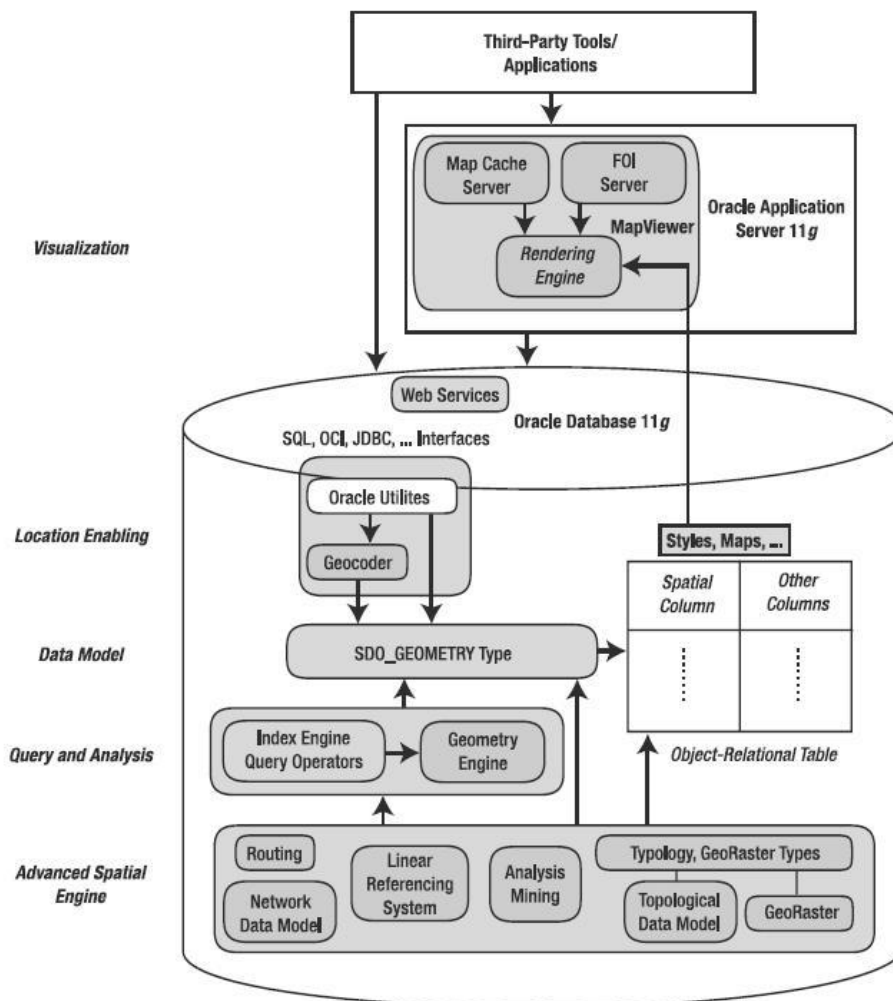


Figure 5-1: Oracle Spatial technology components (Source Kothuri et al., 2007, Page 20)

Without the Oracle Spatial component, geocoding in Oracle is impossible. Thus, it is very important to understand underlying concept and functionality of every Oracle Spatial component, especially Location Enabling. We will discuss each of them separately, because every of these components are in some way affected or have deep connection with geocoding.

5.1 Data Model Component

To store spatial data in Oracle Database, Oracle Spatial uses SQL data type `SDO_GEOMETRY`. With this data type users can define tables with columns to store spatial locations and spatial extents of geographic entities such as railways, buildings, land parcels, etc (Kothuri, et. al, 2007). We discuss `SDO_GEOMETRY` data type in the following chapter in detail, because it covers large spectrum of study, and is beyond of the purpose of this chapter.

5.2 Location enabling Component

Location-enabling is a process of enabling functional and geographic referenced spatial data within database. To do spatial query and analysis, or to perform network routing and analysis, or to visualise our data within Oracle Database using Oracle Spatial component, first we need to location-enable our application. We can perform location-enabling with Oracle Spatial in several ways: by adding `SDO_GEOMETRY` columns to application tables, or by populating tables with this data type using loading or importing system. If, for example, we consider a business-application that stores information about its customers, we have to location-enable it to be able to perform various types of spatial analysis, or to visualise customer data in meaningful map.

For the process of geocoding, this is the most important component. It is not possible to geocode data, without “location enabled” column where geocoded data are to be stored. Thus, to perform geocoding, we first “location enable” our table and then perform geocoding. Also, Oracle geocoder is a part of this component.

Location in Oracle Spatial is most commonly derived from address component in application tables such as customers, branches, etc, and is stored as a point location in these tables. In listings 5-1 and 5-2, I have provided simple examples how to location-enable new table within CREATE statement and using SDO_GEOMETRY data type.

```
CREATE TABLE schools_of_san_francisco (  
  id NUMBER,  
  
  name VARCHAR(30),  
  location SDO_GEOMETRY  
);
```

Listing 5-1: Location-enabling process for schools_of_san_francisco

```
CREATE TABLE universities_of_san_francisco (  
  id NUMBER,  
  university_name VARCHAR(100),  
  geom SDO_GEOMETRY  
);
```

Listing 5-2: Location enabling process for universities_of_san_francisco

5.3 Spatial Query and Analysis Component

This component provides the core functionality for querying and analysing spatial geometries (Kothuri, et. al, 2007). It is constructed from two subcomponents: Geometry Engine and Index Engine. Geometry Engine provides functions to analyse, manipulate and compare geographical data, while Index Engine provides possibility to create an index on spatial data (Kothuri, et. al, 2007).

If we presume that we have created schools and university tables as shown in listings 5-1 and 5-2, and that we have populated them with data, we can do spatial

querying as shown in Listing 5-3, where five schools are selected with nearest location to the University of San Francisco.

```
SELECT name
FROM
(
  SELECT name,
  SDO_GEOM.SDO_DISTANCE(P.location, G.geom, 0.5) distance
  FROM universities_of_san_francisco G,
  schools_of_san_francisco P
  WHERE G.university_name = 'University of San Francisco'
  ORDER BY distance
)
WHERE ROWNUM <=5;
```

Listing 5-3: Example of spatial query using SQL with location-enabled data

5.4 Advanced Spatial Engine Component

Advanced Spatial Engine is a part of Oracle Spatial which comprises several components that cater to sophisticated spatial applications. For example, we can perform routing or spatial analysis of geocoded information. Its main purpose is complex analysis and complex manipulation of spatial data as required in traditional GIS. Table 5-1 presents several Advanced Spatial Engine components and explains their purpose.

Table 5-1: Example list of Advanced Spatial Engine components

Component	Functionality
Network Data Model	Provides a model for storing network data
Linear Referencing System	Translation of markers of linear features to geographic coordinate space and vice versa

Spatial Analysis and Mining Engine	Combine and analyse geographic and demographic data
GeoRaster	Functionality needed to work with, par example satellite images and to store them into Oracle Database
Topology Data Model	Analysis and manipulation of spatial geometry data using finer topological elements

Every of these components of Advanced Spatial Engine use already mentioned geometry data type, Index Engine and Geometry Engine functionality.

5.5 Visualisation Component

With Oracle Spatial technology we are able to visualize spatial data via MapViewer tool (Figure 5-2). This tool renders spatial data that are stored in SDO_GEOMETRY column and can present them as the map. Thus, when we geocode address information, we are able to present and visualise them. When speaking about visualisation, Oracle Spatial as well provides Oracle Map suite of technologies, which main purpose is enabling fast map browsing using map cache server and feature of interest (FOI) server. Oracle Map suite uses cache server that pregenerates maps and caches image tiles for a map, while FOI server renders dynamic application content for spatially enabled tables using a combination of images and geometric themes (Kothuri, et. al, 2007).

In addition to vector data saved in SDO_GEOMETRY table columns, MapViewer can as well display spatial data stored in image (raster) format. Raster data are stored in tables using SDO_GEORASTER data type.

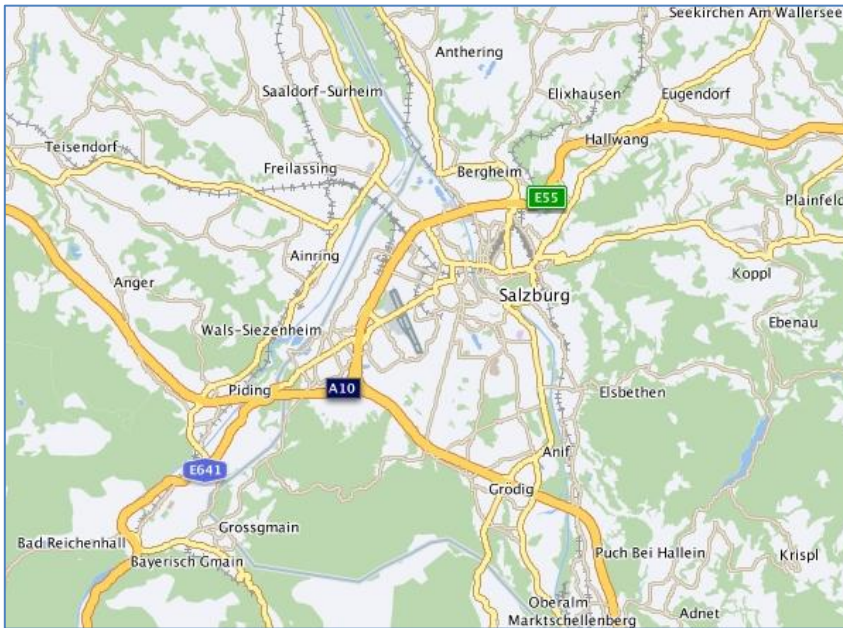


Figure 5-2: A sample map with multiple themes generated using MapViewer.

6 SDO_GEOMETRY Data Type

In a database terminology, objects that contain data have an associated data type that defines the kind of data. SDO_GEOMETRY data type in Oracle Spatial component is used to store spatial data into Oracle Database. This type of data allows us to save geographic information about roads, parks, state boundaries and many more, and gives us possibility to use this data in spatial analysis or visual representation of them on the map. In addition, it makes possible and feasible to perform geocoding within Oracle and to store geocoded data in appropriate manner.

6.1 Spatial geometries in Oracle

The SDO_GEOMETRY data type is used for storing and modelling of various geometries as: point, line, polygon and even complex geometries. Because of this, we have a huge number of available possibilities to store our data within Oracle Database. Figure 6-1 shows an example of the types of the spatial data that SDO_GEOMETRY can store.












Supported Types in 2D, and 3D (F,G not supported in 3D)	<div>A  Point</div> <div>B  Line String</div> <div>C  Polygon (Area)</div> <div>D  Polygon with a Hole</div> <div>E  Collection</div> <div>F  Compound Line String</div> <div>G  Compound Polygon</div>
3D-only Types	<div>H  Composite Surface</div> <div>J  Simple Solid</div> <div>K  Composite Solid</div> <div>L  Collection</div>

Figure 6-1: Example of spatial data that SDO_GEOMETRY can represent (Source Kothuri, et al., 2007, Page 56).

6.1.1 Points

The simplest geometry that can be saved in SDO_GEOMETRY data type is point. During the implementation of geocoding in this work, we will mostly work with the points. This type of geometry can be used to present certain location of clients, customers, competitors, store locations, etc. To store point location in SDO_GEOMETRY we have to provide at least two coordinates (x and y). Object A in Figure 6-1 is an example of Point geometry. Simple example of point representation on the map is provided in Figure 6-2, where Salzburg city and surrounding municipalities are presented as points with x and y coordinates.

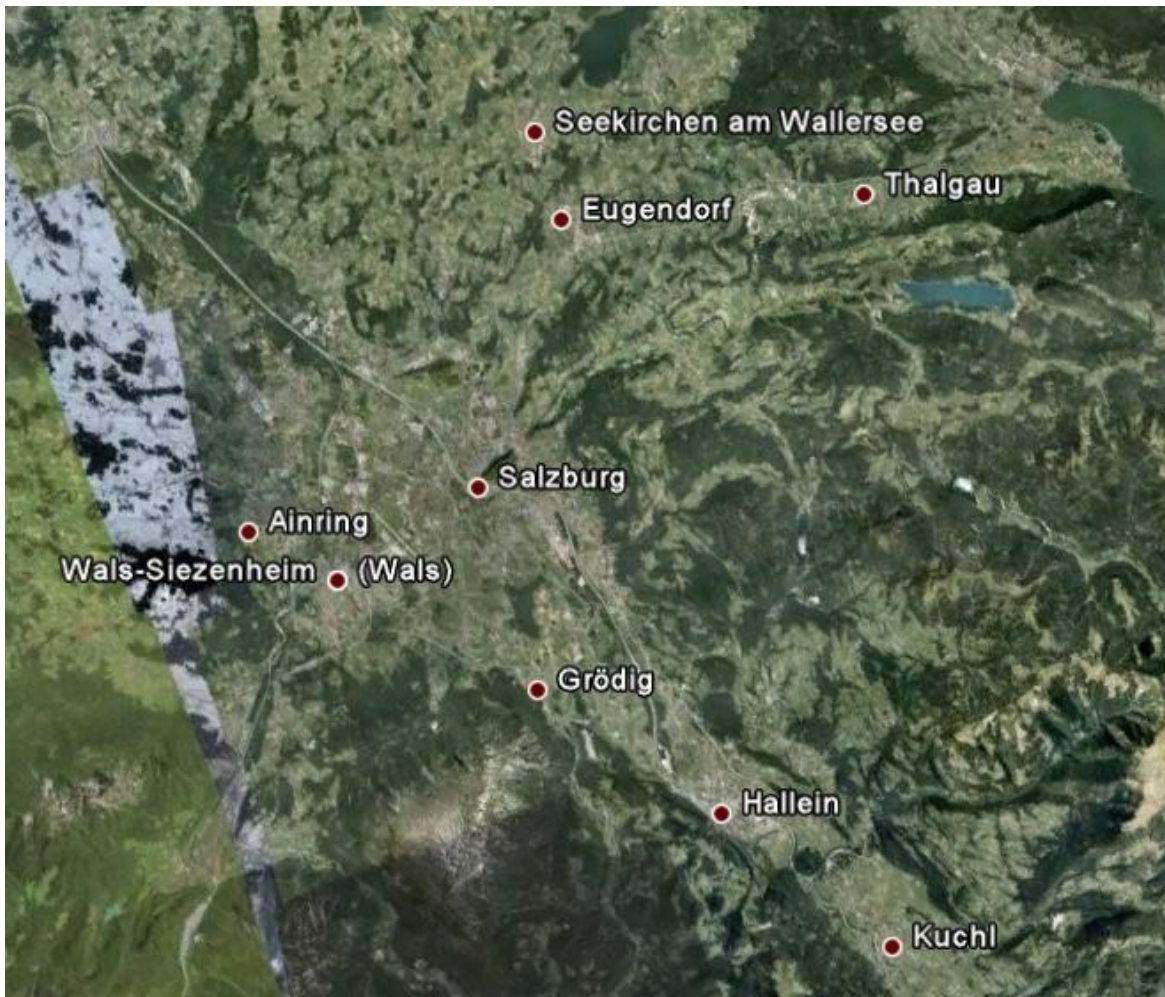


Figure 6-2: Point representation of the cities surrounding Salzburg city (Source: Google Earth TM).

6.1.2 Line String

Line String in GIS terminology is defined as connection of multiple points, or sometime called vertices. We use this type of data to store and retrieve information about roads, routes, railways, pipelines and many others. In SDO_GEOMETRY this kind of data is represented as Line String. If Line String is closed, we define it as Ring. Line String connects two or more points as straight lines where there is no ambiguity; or as circular arcs; or as a combination of straight lines and circular arcs (Kothuri, et. al, 2007). Objects B and F in Figure 6-1 represent various types of Line String.

6.1.3 Polygon and Surfaces

Polygon is a plane figure that is bounded by a closed path or polyline, composed of a finite sequence of straight line segments. Polygon can have any arbitrary shape and as such we can use it to represent areas, buffer zones, shapes of the countries, etc. This type of object is as well supported by SDO_GEOMETRY and in Figure 6-1 is represented as objects C, D and G. The simple example of polygons is presented in Figure 6-3, where countries of Europe are presented as polygons.



Figure 6-3: Countries of Europe presented as polygons

6.1.4 Solids

Difference between polygon (2D) and solid (3D) are: solid is constructed from area and volume, while polygon is constructed only from area. Object J in figure 6-1 is a simple example of solid. In most cases, when we want to present some object in 3D on Google Earth for example, we use solids, but composite. Composite solid is constructed from collection of simple solids that have a single volume. Object K in Figure 6-1 is an example of composite solid created from two simple solids.

6.1.5 Collections

Multiple geometries collected as one element are called “collections” in Oracle Spatial. They could be heterogeneous as combination of points, line and/or polygons, or they can be homogeneous which mean they consist of elements of a single type. Object E in Figure 6-1 is an example of collection and it has two polygons presented as collection. In most cases when speaking about map presentation, collections are used to present countries. For example, in the case of Croatia (Figure 6-4), to present a complete country as one entity we use collections of polygons. One polygon is used to present continental part of Croatia, while other polygons are used to present islands.



Figure 6-4: Blank map of Croatia

6.2 Attributes and their values of SDO_GEOMETRY data type

SDO_GEOMETRY is a data type that enables storage and manipulation of spatial data within Oracle Spatial component in Oracle Database. This data type has several attributes which we obtain with use of SQL query `DESCRIBE` (Figure 6-5). This is a data type that is used by Oracle geocoder to store geographical

coordinates. Thus, it is important to know its functionalities and working environment when dealing with geocoding.



Figure 6-5: Use of DESCRIBE SQL statement to retrieve information about SDO_GEOMETRY

After execution of the query in Figure 6-5, our result looks like in Figure 6-6.



Figure 6-6: SDO_GEOMETRY data type in Oracle

6.2.1 SDO_GTYPE Attribute

This attribute of SDO_GEOMETRY describes the type of geometric shape in the object. The main purpose of SDO_GTYPE is to indicate whether the geometry is arbitrary collection, multipoint, multipolygon, multiline, line string, polygon or a point. It can be a combination of multiple elements where every of them has different shape. In any case, this attribute defines general type for an entire object. It consists of a combination of four numbers structured as D00T. Only the first and the last digit are changeable and can take different values. The first number in D00T represents dimension of the geometry while the last number defines shape or type of geometry. So, for example, representation of a two-dimensional point

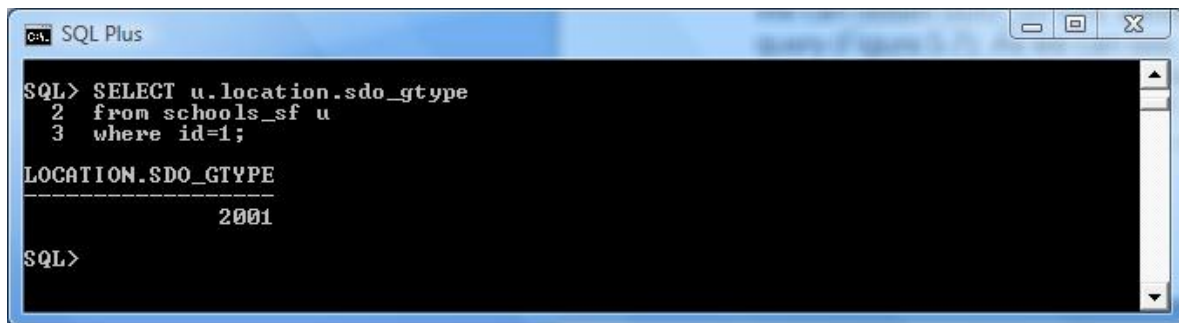
D00T looks like: 2001. To represent a three-dimensional polygon, D00T looks like: 3003. Possible values for dimension of D00T format are presented in Table 6-1.

Oracle Spatial can work with two, three and four-dimensional objects. If an object has two-dimensional characteristic then each vertex has two ordinates. If an object has three-dimensional characteristic then each vertex has three ordinates, and so on (Kothuri, et al., 2007).

Table 6-1: Values of D00T Format of the SDO_GTYPE attribute

DIGIT	VALUES
D (dimension of the geometry)	2 = Two-dimensional 3 = Three-dimensional 4 = Four-dimensional
T(shape or type of geometry)	0 = Uninterpreted type 1 = Point 2 = Line 3 = Polygon /Surface 4 = Collection 5 = Multipoint 6 = Multiline 7 = Multipolygon / Multisurface 8 = Solid 9 = Multisolid

We can obtain SDO_GTYPE attribute for a certain object in a database using SQL query (Figure 6-7). As we can see, this object is two-dimensional because the first number in the result of the query is 2, and has a point type of geometry because the last number in D00T result is 1.

The image shows a screenshot of an SQL Plus window. The title bar says "SQL Plus". The command window contains the following text:

```
SQL> SELECT u.location.sdo_gtype  
2   from schools_sf u  
3  where id=1;  
  
LOCATION.SDO_GTYPE  
-----  
                2001  
  
SQL>
```

Figure 6-7: Example of the SDO_GTYPE in location column of schools_sf table

6.2.2 SDO_SRID Attribute

This attribute of SDO_GEOMETRY represents and stores information about spatial reference system or coordinate system for the geometry. Spatial Reference System Identifier (SRID) is a unique value used to identify spatial coordinate system. It is always represented as integer value. As can be seen in Figure 6-8, resulted SRID of our query over `schools_sf` table returned integer values of – 8307. It means that object is defined with coordinate system whose SRID is 8307 and whose well-known name is "Longitude / Latitude (World Geodetic System 84 (WGS 84))".

This is probably the most widely used coordinate system, and it is the one used for global positioning system (GPS) devices. The geometries are then transformed using the coordinate system whose SRID is 8199 and whose well-known name is "Longitude / Latitude " (Murray 2003).

SRID is very important in the process of geocoding and application of geocoded results, because it represents coordinate system of the data. Defining SRID of referenced data during geocoding, we define coordinate system of our resulting set.



```
SQL> SELECT u.location.sdo_srid
2   from schools_sf u
3  where id=1;

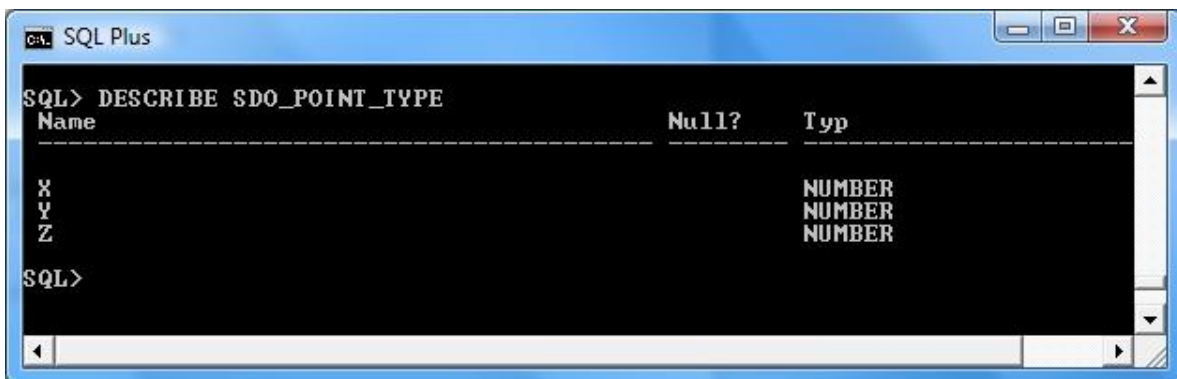
LOCATION.SDO_SRID
-----
                8307

SQL>
```

Figure 6-8: Example of SRID values for schools_sf table

6.2.3 SDO_POINT Attribute

This attribute specifies location of the point geometry. Because, work in this master thesis is based on location geocoding, which are represented as points (2D data), this is important attribute. An example is a location of a client, customer or store. This attribute additionally is a type of SDO_POINT_TYPE, which is another object type as well. The SDO_GTYPE for point geometry is set to D001. This attribute can store only three ordinates (Figure 6-9): x, y and z. For additional ordinates we have to use SDO_ELEM_INFO and SDO_ORDINATES attributes.



```
SQL> DESCRIBE SDO_POINT_TYPE

Name          Null?    Typ
-----
X              NUMBER
Y              NUMBER
Z              NUMBER

SQL>
```

Figure 6-9: SDO_POINT_TYPE Data type

6.2.4 SDO_ELEM_INFO and SDO_ORDINATES

These attributes are used when we need to store elements more complex than points. SDO_ELEM_INFO and SDO_ORDINATES allow us to specify different elements that compose geometry. SDO_ELEM_INFO specify where element starts

in SDO_ORDINATES and type, while SDO_ORDINATES store coordinates of the vertices in all elements of geometry. Because, this work is about geocoding of the points, these two attributes would not be studied and explained more detailed.

7 Geocoding with Oracle Spatial

An explanation of geocoding in general is provided in chapter two. In this chapter, focus is geocoding with Oracle Spatial. Here, we present information required to conduct the process of geocoding in Oracle. Oracle Geocoder architecture, geocoding process flow and main geocoding functions will be described and evaluated.

7.1 Oracle Geocoder

Oracle geocoder is a collection of functionalities in Oracle Spatial to perform process of geocoding. Such functionalities include address parsing, search and cleaning (address normalization) and coordinates generation. Result of the work of the geocoder is output of corrected address in case of need and derived geographical coordinates for input data without previous spatial reference.

7.1.1 Parsing

To perform action with geocoder and to use its functionality, first we need reference data such as the addresses with known geographical coordinates. Additionally, we need input data that have no spatial reference to parse them with geocoder. As can be seen in Figure 7-1, geocoder first parses this data to recognizable elements. The purpose of this part of the geocoding process is to recognize the parts of the street address and separate its elements for easier searching and recognition. Even so this part could be very problematic, because of the various standardizations of addresses from country to country, geocoder can recognize variety of addresses. We can find out what countries and what kind of addresses can be recognized using table GC_PARSER_PROFILEAFS in Oracle Spatial in reference data.

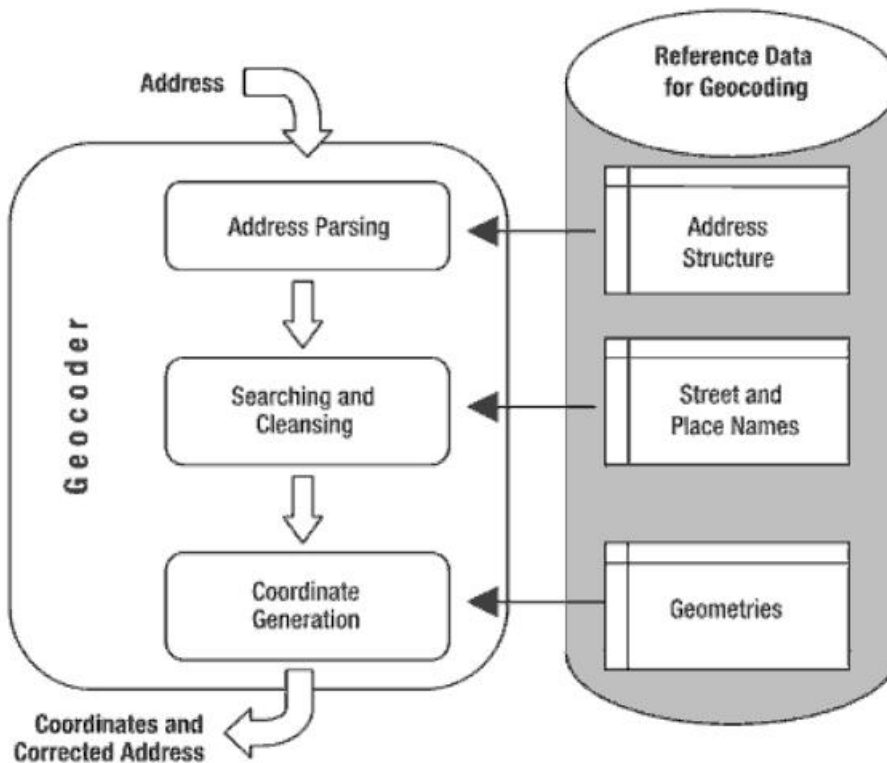


Figure 7-1: Oracle geocoder architecture Source: (Kothuri, et al., 2007, Page 153)

7.1.2 Address Search

After successful parsing, geocoder comes to the stage of searching for an address with matching name in a database that most closely matches the given input address. This search can be considered as fuzzy, because geocoder can find its match even if input address is misspelled. If specified address cannot be found, geocoder falls back to postal code of the address or to city name. However, user will have possibility to specify if this action is acceptable via matching mode parameter (Kothuri, et. al, 2007). The most important thing regarding parsing is that at the end, user will have a collection of cleaned addresses and with correct formatting. Also, missing elements will be added during the process of parsing like postal code or a part of the street name.

7.1.3 Coordinates computation

After search was performed and a proper street was found, geocoder converts input data into a geographical point. The output quality of this process, however, depends on the correlation between observed points in reference data. To explain this, first I need to say that Oracle geocoder only holds the house numbers at each end of the street and, geocoder computes the geographical location of the input point of interest by interpolation. It means, if we want to derive an address and geographical location of the point of interest, for example Sutro Elementary School in San Francisco, which is located somewhere in the middle of a street, geocoder interpolates coordinates of this point according to the position of the objects at each end of that street, because it assumes that houses are regularly spaced along linear geometry that represent the street segment. Figure 7-2 illustrates result of this process, where number “6” is assigned to Sutro Elementary School, according to numbers at the each end of the street. So, if houses are evenly distributed along the street, output result will be quite good. Otherwise, if the objects of the street do not have equal spatial distribution, results can be very erroneous. As said before, it is because geocoder assumes that objects are equally distributed along the street. This system is useful and accurate for calculation of data within organized municipalities and cities like in USA and some parts of the Europe. Output addresses as a product of geocoding are always in the coordinate system used by reference data

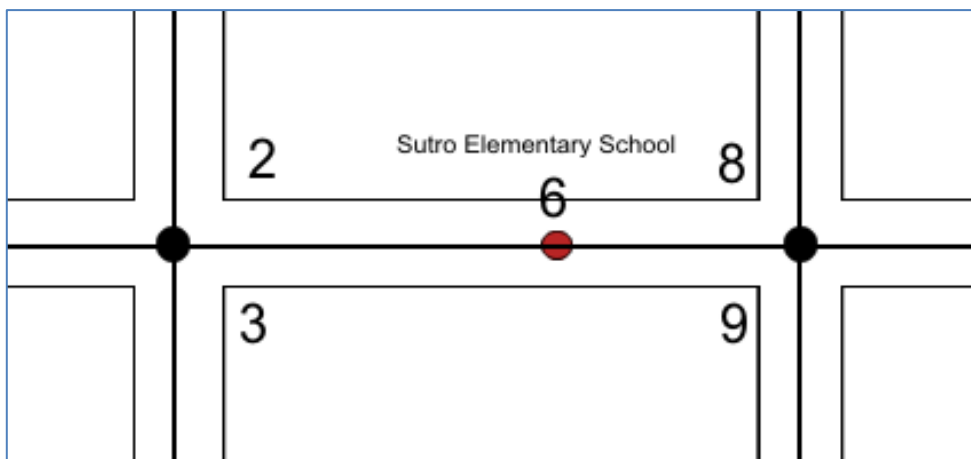


Figure 7-2: Interpolation example

7.2 Reference data

Reference data are used as starting point for determining geographic coordinates of input data. Reference data can be considered as “data with instructions” for the purpose of geocoding. We use a coordinate system of these data and their sample points to achieve geocoding functionality using interpolation method in Oracle Spatial. Reference data in Oracle Spatial are implemented as a set of tables with specific structure and all of them have a prefix in the name like GC_. Reference data tables, can be divided them into two categories: parameter tables and data tables.

7.2.1 Parameter tables

These tables are used for control of operation performed by geocoder. They contain information about structuring of addresses in each country which is supported by geocoder. The list of parameter tables with description of their purpose is presented in Table 7-1.

Table 7-1: Parameter tables supported by Oracle geocoder

TABLE NAME	DESCRIPTION
GC_COUNTRY_PROFILE	In this table, information about every country that is known to Oracle geocoder is stored, e.g. administrative levels and political construction of a country.
GC_PARSER_PROFILEAFS	This table in addition to table GC_COUNTRY_PROFILE contains information about structure of the address for each country supported by Oracle geocoder. A country is presented per one row and structure of the address is defined in XML notation.
	This table is used by Oracle geocoder for

GC_PARSER_PROFILES	recognition of elements of an address. Definition of them together with their synonyms and possible misspellings are saved here.
---------------------------	--

7.2.2 Data tables

Data tables contain the place names and their spatial reference. Each of these tables has a name with country-suffix for easier recognition and easier manipulation of data. Country specific suffix is constructed by two letters common as abbreviation for certain country. For example, United States are marked with suffix US, while Germany is identified with suffix DE, Austria with suffix AT, Bosnia and Herzegovina with suffix BA, Croatia with suffix HR, France with suffix FR, etc. Table 7-2 shows the names of data tables and their descriptions. Under column TABLE NAME, xx represent country specific suffix.

Table 7-2: Data tables supported by Oracle geocoder

TABLE NAME	DESCRIPTION
GC_AREA_xx	Information about administrative and political levels of specific country are saved in this table. Within Oracle geocoder, three levels of administrative areas are used: Region, Municipality, and Settlement. How these areas are mapped, it varies from country to country.
GC_POSTAL_CODE_xx	This table describes all postal codes. In addition, it contains the coordinates of a central point for each postal code.
GC_POI_xx	In this table, we can find various collections of points of interest connected with countries defined with suffix. The amount of data which can be found in these table depends on suppliers of data
GC_ROAD_xx	This table is used for address search during the process of geocoding. One row in this table

	presents one road per settlement and postal code. If a road (as in large cities) crosses multiple postal codes, it will appear more times in the table.
GC_ROAD_SEGMENT_xx	This table contains one row for each segment of the road in table and provides information required to compute coordinates of an address by interpolation method.
GC_INTERSECTION_xx	Every intersection of a road is saved in this table. A new row in this table is defined whenever road segments meet at same location.

7.3 Geocoder functions

Geocoding API is simple and it is compound of the PL/SQL package (SDO_GCDR) with only few functions. These functions accept address as an input and calculate geographical coordinate information as a result. The only difference between functions within geocoder are input data and amount of data they return or produce (Kothuri, et al., 2007), as presented in Table 7-3. Additionally in Table 7-4, we can see that every of these functions can do address conversion. In case of address correction functionality, only GEOCODE_AS_GEOMETRY is not capable to execute this function.

Table 7-3: Comparing the Oracle geocoder functions regarding their input and output data structure

Function	Input data	Returned data
GEOCODE_AS_GEOMETRY	Structured addresses, but known to be valid.	<ul style="list-style-type: none"> - Geometric points - No indication of precision - No indication of quality of the results
GEOCODE	Unstructured address, passed as	<ul style="list-style-type: none"> - Returns geographical coordinates and a

	set of address lines	corrected address - Returns detailed indication of the quality of a result set
GEOCODE_ADDR	Uses a structured address	- Same as GEOCODE
GEOCODE_ALL	End user choice of the correct address match	- Like GEOCODE, but additionally returns multiple matches
GEOCODE_ADDR_ALL	Uses a structured address	- Like GEOCODE_ALL

Table 7-4: Comparing the Oracle Geocode functions

Function	Input data	Returned data
GEOCODE_AS_GEOMETRY	Structured addresses, but know to be valid.	- Geometric points - No indication of precision - No indication of quality of the results
GEOCODE	Unstructured address, passed as set of address lines	- Returns geographical coordinates and corrected address - Returns detailed indication of the quality of result set
GEOCODE_ADDR	Uses a structured address	- Same as GEOCODE
GEOCODE_ALL	End user choice of the correct address match	- Like GEOCODE, but additionally returns multiple matches

GEOCODE_ADDR_ALL	Uses structured address	- Like GEOCODE_ALL
-------------------------	-------------------------	--------------------

7.3.1 GEOCODE_AS_GEOMETRY

This function is the simplest function to use in Oracle geocoder. This function converts an address, but it does not correct it. In Listing 7-1, syntax of this function with its parameters is presented: (username, addr_lines, country).

```
SDO_GCDR.GEOCODE_AS_GEOMETRY (
    username          IN VARCHAR2,
    add_lines         IN SDO_KEYWORDARRAY,
    country           IN VARCHAR2
) RETURN SDO_GEOMETRY;
```

Listing 7-1: Syntax of GEOCODE_AS_GEOMETRY function

7.3.1.1 *username*

This is the name of the Oracle schema that contains geocoding tables for specific country and it is required argument to perform geocoding process (Kothuri, et al., 2007).

7.3.1.2 *addr_lines*

This is a simple array of character strings that we use to pass address lines to any geocoding function in Oracle and it is type of SDO_KEYWORDARRAY. Addresses must match structure described in GC_PARSER_PROFILEAFS and formatted properly. However, there are some possibilities of flexibility in their formatting which are presented in example listings 7-2, 7-3 and 7-4. However, a street with its number must be submitted on a separate line.

```

SDO_KEYWORDARRAY (
  '3333 Example Street'      -- Street
  'City of One'              -- City
  'AA'                       -- State
  '10000'                    -- Postal code
)

```

Listing 7-2: Example of street, city, state abbreviation and postal code on separate lines

```

SDO_KEYWORDARRAY (
  '3333 Example Street'      -- Street
  'City of One'              -- City
  'AA 10000'                 -- State and postal code
)

```

Listing 7-3: Example of state abbreviation and postal code in one line

```

SDO_KEYWORDARRAY (
  '3333 Example Street'      -- Street
  'City of One AA 10000'     -- City, state and postal code
)

```

Listing 7-4: Example of city, state abbreviation and postal code in one line

7.3.1.3 *country*

This is a two-letter International Organization for Standardization (ISO) code for a country whose addresses we intend to geocode belong to.

7.3.2 GEOCODE

As we already can see in Tables 7-3 and 7-4, this function in addition to GEOCODE_AS_GEOMETRY does address correction during the process of

geocoding. Not only that, GEOCODE as a result of geocoding process, in addition to geographical coordinates, returns a fully formatted address and codes that tell us precisely how the address matched (Kothuri, et al., 2007). It contains same parameters such as GEOCODE_AS_GEOMETRY, but in addition it has MATCH_MODE parameter, which determines how closely the attributes of an input address must match the data being used for the geocoding.

7.3.2.1 match_mode

Input addresses can represent the same thing in different way, such as Street and the abbreviation St or Str. They can include minor errors, such as wrong postal code. With match_mode parameter, during the process of geocoding, we can require an exact match, or relax the requirements for some attributes between the input address and the data used for geocoding. A complete overview of match mode is presented Table 7-5.

Table 7-5: Match modes and their descriptions

Match Mode	Description
EXACT	All attributes of the input address must match the data used for geocoding
RELAX_STREET_TYPE	The street type can be different from the data used for geocoding
RELAX_POI_NAME	The name of the point of interest does not have to match the data used for geocoding
RELAX_HOUSE_NUMBER	The house number and street type can be different from the data used for geocoding
RELAX_BASE_NAME	The base name of the street, the house number, and the street type can be different from the data used for geocoding.
RELAX_POSTAL_CODE	The postal code, base name, house number, and street type can be different from the data used for geocoding
	The address can be outside the specified city specified, but

RELAX_BUILTUP_AREA	must be in same county. Additional characteristics of RELAX_POSTAL_CODE.
RELAX_ALL	= RELAX_BUILTUP_AREA
DEFAULT	= RELAX_BASE_NAME

7.3.2.2 MATCHCODE

While match_mode parameter allows us to define how closely the attributes of an input address must match the data being used for the geocoding, MATCHCODE indicates the way the input address in resulting set was matched with the list of the addresses from reference data. There are three attributes that make possible to find mistakes or errors during the process of geocoding: MATCHCODE, ERRORMESSAGE, and MATCHVECTOR. In this thesis, we focus on only MATCHCODE, because of the lack of the time. Other two attributes are irrelevant for the purpose of the result analyse in this work. MATCHCODE is the number that indicates which input address attributes matched the data used for geocoding and it is stored in the MATCH_CODE attribute of the output SDO_GEO_ADDR object; i.e. it indicates quality of a derived result set.

Table 7-6 gives us overview of all match codes and their descriptions regarding information on matching within reference data.

Table 7-6: Match codes for geocoding operations and their description

Match code	Description
1	Exact match.
2	Everything match, except street type, suffix, or prefix.
3	Everything match, except house or building number.
4	The city name and postal code match, but the street address does

	not.
10	The city name matches, postal code does not.
11	The postal code matches, but not the city name.

7.3.3 GEOCODE_ADDR

This function is identical to GEOCODE, but it takes SDO_GEO_ADDR object as an input, instead SDO_KEYWORDARRAY. Parameters “country” and match_mode are submitted as a part of SDO_GEO_ADDR object.

The syntax of this function is provided in Listing 7-5, and an example of calling the GEOCODE_ADDR function from dummy table DUAL is provided in Listing 7-6.

```
SDO_GCDR.GEOCODE_ADDR
(
    username      IN VARCHAR2,
    address       IN SDO_GEO_ADDR
)
RETURN SDO_GEO_ADDR;
```

Listing 7-5: Syntax of SDO_GEO_ADDR function

```
SELECT SDO_GCDR.GEOCODE_ADDR
(
    'SPATIAL'           -- schema name
    SDO_GEO_ADDR        -- object
    (
        'US',           -- country ISDO code
        'DEFAULT',      -- matchmode
        '1111 Example Street', -- street
        'City of One',  -- settlment
        NULL,           -- municipality, if any
    )
)
```



```
'CA',          -- state or region
'10000'       -- postal code
)
FROM DUAL;
```

Listing 7-6: Example of calling the GEOCODE_ADDR function from dummy table DUAL

7.3.4 GEOCODE_ALL and GEOCODE_ADDR_ALL

GEOCODE_ALL function is used when we have the so called ambiguous input data that can produce results with multiple matches. An example of an ambiguous address is “45 Bahnhof, Wien, 1000”, because it can produce several matches like: “45 Bahnhofstrasse, Wien, 1000”, or “45 Bahnhofgasse, Wien, 1000”, or even “45 Bahnhofweg, Wien, 1000”. This function returns only one match for ambiguous addresses and lets the end user select which of these addresses is a correct match.

It contains same parameters as GEOCODE, but instead of returning single match in SDO_GEO_ADDR object, it returns an array of SDO_GEO_ADDR objects as an object of type SDO_ADDR_ARRAY (Kothuri, et. al, 2007).

GEOCODE_ADD_ALL is a function which is identical to GEOCODE_ALL, but as an input it takes SDO_GEO_ADDR object – not SDO_KEYWORDARRAY. In this thesis we will not work with these two functions because of several reasons: these functions return more than one match in result set and that is not a focus of our research; our sample set is not adequate for the purpose of evaluation of these functions, because it contains information about street in full format; evaluation of these functions requires too much time

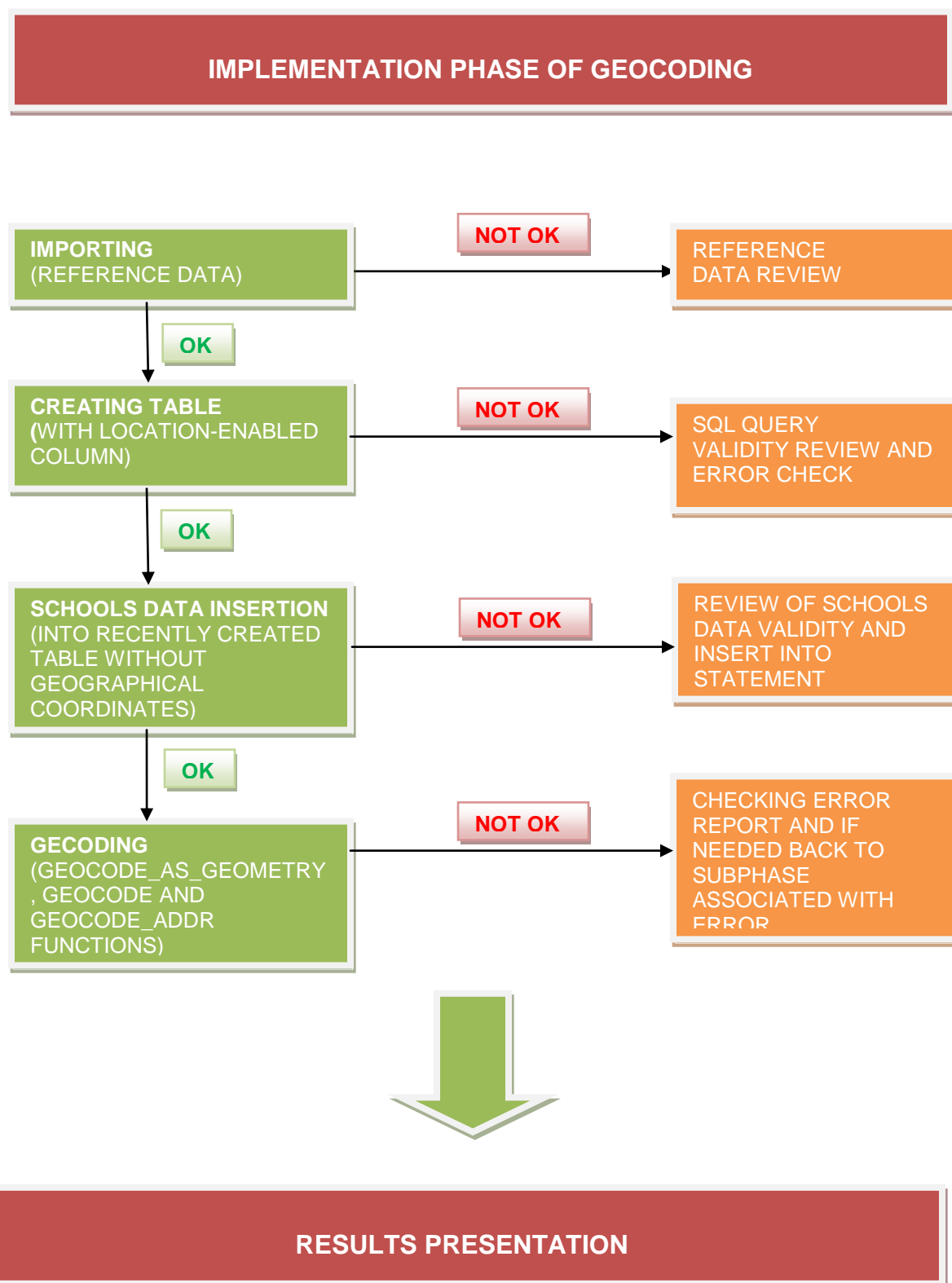
8 Implementation of geocoding and its accuracy assessment

In this chapter I implement geocoding process within Oracle and evaluate three Oracle Spatial geocoder functions which return one matching value for every address as a result. In addition, accuracy assessment for the resulting set is provided. For the purpose of the testing, I have collected ten simple addresses of schools in the City of San Francisco without geographical coordinates as can be seen in Appendix A. The main aim here is to explain, how already saved data in Oracle database and without geographical coordinates, can be used in conversion and retrieving spatial referenced information.

8.1 Geocoding with GEOCODE_AS_GEOMETRY

To execute this function, we need to have an address set for which we know that addresses are valid. Address validity is not a problem with our sample, because all of the addresses were checked and validated several times. Addresses and names of the schools are collected manually using Google Earth and Yahoo Maps during October 2010, and validated through prospective websites of the schools and various school directories of the state of California. One of the main geocoding challenges is to provide good and valid reference data (Arctur and Zeiler 2004). Thus, as a reference file for geocoding purposes, I obtained `gc.dmp` file from `apress.com` website, which contains geocoding data for two cities in United States: Washington D.C. and San Francisco. This data are validated and presented in the work of Kothuri et. al (2007).

To easier understand the process of geocoding implemented here, presentation and analysis of the results, I have created workflow diagram that can be seen as a Figure 8-1. “Implementation phase of geocoding” in diagram represents universal approach to geocoding regardless of which function of Oracle geocoder we use. Thus, we use this diagram to present geocoding workflow for every function used in this work.



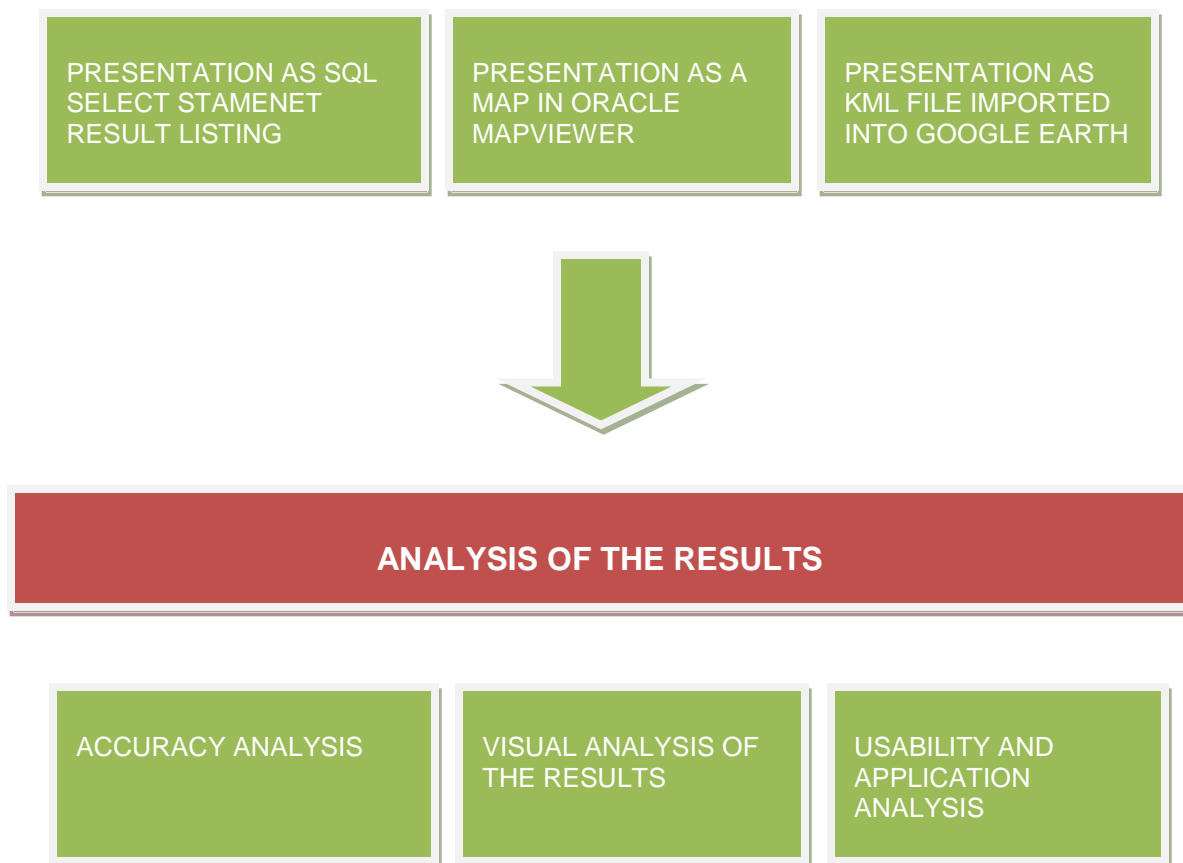


Figure 8-1: Workflow diagram illustrating the process of geocoding and analysis of the results

8.1.1 Implementation

After importing `gc.dmp` file, table of San Francisco schools is created using SQL as can be seen in Listing 8-1; the table name is `schools_sf`, with columns: `id`, `name`, `addr_number`, `street`, `postal_code`, `city`, `state_abrv` and `location`.

```
CREATE TABLE schools_sf(  
    id NUMBER,  
    name VARCHAR(100),  
    addr_number VARCHAR(10),  
    street VARCHAR(50),  
    postal_code VARCHAR(5),  
    city VARCHAR(20),  
    location
```

```
state_abrv VARCHAR(2),  
location SDO_GEOMETRY );
```

Listing 8-1: CREATE TABLE statement for creating San Francisco table in Oracle database

After creating a table, I arranged data from Appendix I (contains address data about ten schools in City of San Francisco) and used them in SQL statement for inserting data into a table. Complete SQL query can be seen in Appendix II. Data were inserted without `location` information, because we still do not possess them.

After execution of `INSERT` statement, information about San Francisco Schools were saved into table `schools_sf`. In Figure 8-2, there is a preview of

```
SELECT * FROM spatial.schools_sf
```

query done with Oracle Enterprise Manager (OEM), because of finer preview. Note the prefix `spatial` before `schools_sf` in SQL query. It means that correspondent table can be found in `spatial` schema in the relevant Oracle Database instance.

“Location” column, which is `SDO_GEOMETRY` data type, in the resulting set (Figure 8-2) does not contain any data. To produce data to populate “location” column we use `GEOCODE_AS_GEOMETRY` function in Oracle database.

Last Executed SQL							
SELECT * FROM spatial.schools_sf							
Last Execution Details							
SQL Repair Advisor SQL Details Schedule SQL Tuning Advisor							
Results Statistics Plan							
Execution Time (seconds) 0.0090							
ID	NAME	ADDR_NUMBER	STREET	POSTAL_CODE	CITY	STATE_ABRV	LOCATION
1	Sutro Elementary School	235	12th Avenue	94118	San Francisco	CA	
3	Convent of the Sacred Heart High School	2222	Broadway Street	94115	San Francisco	CA	
4	Galileo Academy of Science & Technology	1150	Francisco Street	94109	San Francisco	CA	
5	Sts Peter & Paul School	666	Filbert Street	94133	San Francisco	CA	
2	Bay School of San Francisco	35	Keyes Avenue	94129	San Francisco	CA	
6	Creative Arts Charter School k-8	1601	Turk St	94115	San Francisco	CA	
7	The Urban School of San Francisco	1563	Page Street	94117	San Francisco	CA	
8	IHS - French American International School	150	Oak Street	94102	San Francisco	CA	
9	Mission High School	3750	18th Street	94114	San Francisco	CA	
10	Archbishop Riordan	175	Phelan Avenue	94112	San Francisco	CA	
SQL Repair Advisor SQL Details Schedule SQL Tuning Advisor							

Figure 8-2: Preview of SELECT * FROM spatial.schools_sf table with OEM

The further step is to use recently stored data of the San Francisco schools (also visible in Figure 8-2) for the purpose of creating geographical coordinates and associating them with existing data in our schools_sf table. To achieve that, I used the query presented in Listing 8-2.

```
UPDATE schools_sf
SET LOCATION =
(
SELECT SDO_GCDR.GEOCODE_AS_GEOMETRY
(
'SPATIAL',
SDO_KEYWORDARRAY(addr_number || ' ' || street, city || ', '
|| state_abrv),
'US'
)
FROM DUAL);
```

Listing 8-2: Geocoding in Oracle Spatial using GEOCODE_AS_GEOMETRY function

As can be seen, this query updates table of San Francisco schools and sets new data in location column using GEOCODE_AS_GEOMETRY function from existing data (addr_number, street, city and state_abrv).

8.1.2 Result

Now, when we execute `SELECT` statement with `location` column from `schools_sf` table, we have geographical coordinates as a result. In Listing 8-3, we have executed `SELECT` statement for `id`, `name` and `location` column over `schools_sf` table in the database. The SQL result set is tuned with SQLPlus for better and easier understanding. Now, we have spatial information for every school in our database that allow us visualisation with Oracle MapViewer as in Figure 8-3, or to perform spatial analyse.

NOTE: Explanation and description of MapViewer functionality will not be evaluated in this thesis, because this topic covers large spectrum of study and because of lack of the time.

```
SQL> select id, name, location from schools_sf;
```

ID	NAME	LOCATION
3	Convent of the Sacred Heart High School	SDO_GEOMETRY(2001, 8307, SDO_POINT_TYPE(-122.4336, 37.7944029, NULL), NULL, NULL)
4	Galileo Academy of Science & Technology	SDO_GEOMETRY(2001, 8307, SDO_POINT_TYPE(-122.42402, 37.803308, NULL), NULL, NULL)
5	Sts Peter & Paul School	SDO_GEOMETRY(2001, 8307, SDO_POINT_TYPE(-122.41034, 37.8012618, NULL), NULL, NULL)
1	Sutro Elementary School	SDO_GEOMETRY(2001, 8307, SDO_POINT_TYPE(122.47088, 37.7838678, NULL), NULL, NULL)
2	Bay School of San Francisco	SDO_GEOMETRY(2001, 8307, SDO_POINT_TYPE(-122.45634, 37.8002211, NULL), NULL, NULL)
6	Creative Arts Charter School k-8	SDO_GEOMETRY(2001, 8307, SDO_POINT_TYPE(-122.43538, 37.78001, NULL), N


```

                NULL, NULL)

7 The Urban School of SDO_GEOMETRY(2001, 8307, SDO_POINT_
  San Francisco      TYPE(-122.44642, 37.7709808, NULL),
                    NULL, NULL)

8 IHS - French America SDO_GEOMETRY(2001, 8307, SDO_POINT_
  n International Scho TYPE(-122.42175, 37.7750829, NULL),
  ol                  NULL, NULL)

9 Mission High School SDO_GEOMETRY(2001, 8307, SDO_POINT_
                    TYPE(-122.42728, 37.7612437, NULL),
                    NULL, NULL)

10 Archbishop Riordan SDO_GEOMETRY(2001, 8307, SDO_POINT_
                    TYPE(-122.45239, 37.7273505, NULL),
                    NULL, NULL)

10 rows selected.

```

Listing 8-3: SELECT statement for id, name and location column over schools_sf table and returned result set



Figure 8-3: Map presentation of San Francisco schools with MapViewer

In addition to results above, I exported `location` and `name` columns of the every geocoded school as one Keyhole Markup Language (KML) file with transformation to SRID 4326 and presented it in Google Earth. Figure 8-4 shows screenshot of presentation of exported KML file (Appendix IV) and imported into Google Earth.

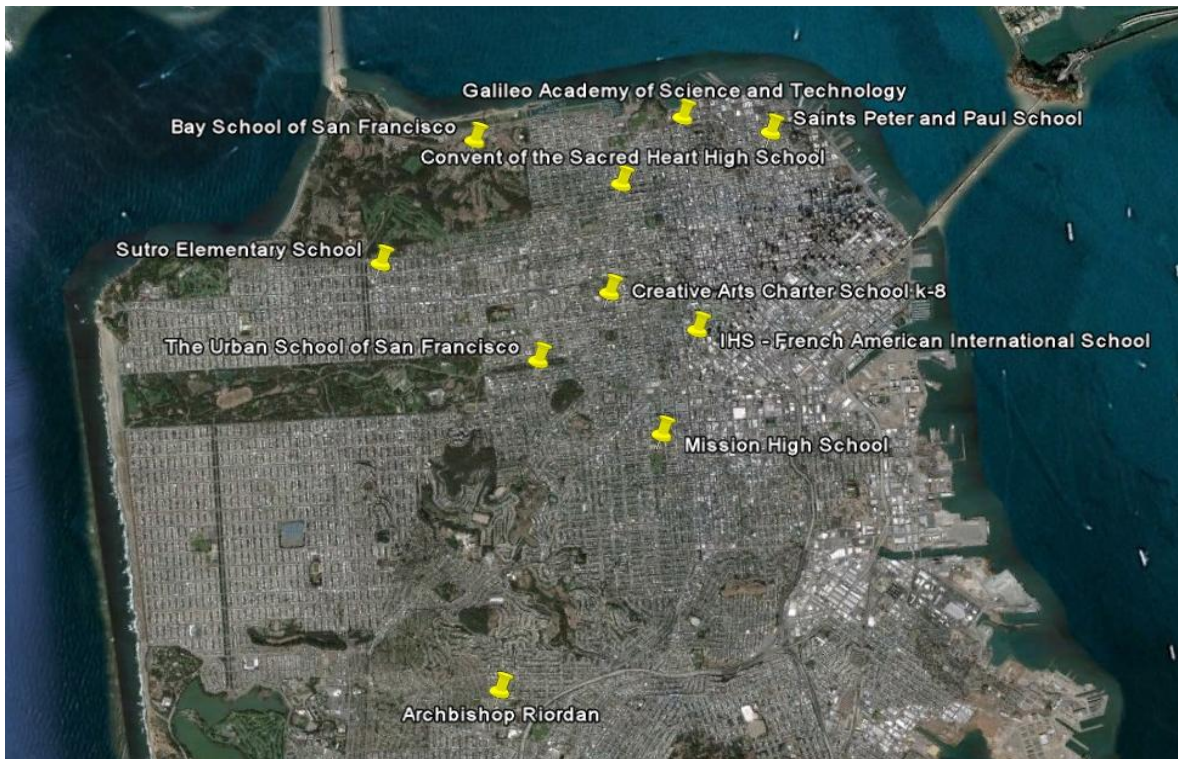




Figure 8-4: Visual presentation of exported KML file in Google Earth

8.1.3 Analysis of the result

Geocoding with this function, besides x and y coordinates, did not produce any additional information about accuracy of the `location` column data, and this can be seen in `location` row of result set in Listing 8-3. `SDO_GEOMETRY` result set for every school, for mentioned listing, gives us only information about a type of geometry (in this case it is point and two dimensional – 2001), SRID (8307 - Note that SRID 8307 and 4326, which I used for creating KML File, are equivalent) and its x and y coordinates values. Additional results about accuracy and validity of addresses are not created by Oracle geocoder using `GEOCODE_AS_GEOMETRY` function.

Even result set can be visualised on the map or serve for the purpose of spatial analysis, acceptable positional accuracy of derived data comes in question when dealing with `GEOCODE_AS_GEOMETRY` function. We define term “acceptable

positional accuracy” regarding map scale for our result set later in this section. To better understand the problem regarding derived result set and their accuracy regarding true position, I have exported `location` and `name` columns of the every geocoded school as separate KML file and imported them into Google Earth. Visual presentation of one school regarding its true position in Google Earth can be seen in Figure 8-5.

As can be determined by visual inspection from Figure 8-5, positional accuracy of the data is not the best. School position geocoded with `GEOCODE_AS_GEOMETRY` function does not coincide with “true location” on Google Earth when dealing with extremely large map scales (in our case map scale is 1:95 in meters). Term “true location” here is default location provided from Google Earth for the observed object. School we used as an example here is “Convent of the Sacred Heart High School” and is marked with red marker , while geocoded location is marked with yellow pin .

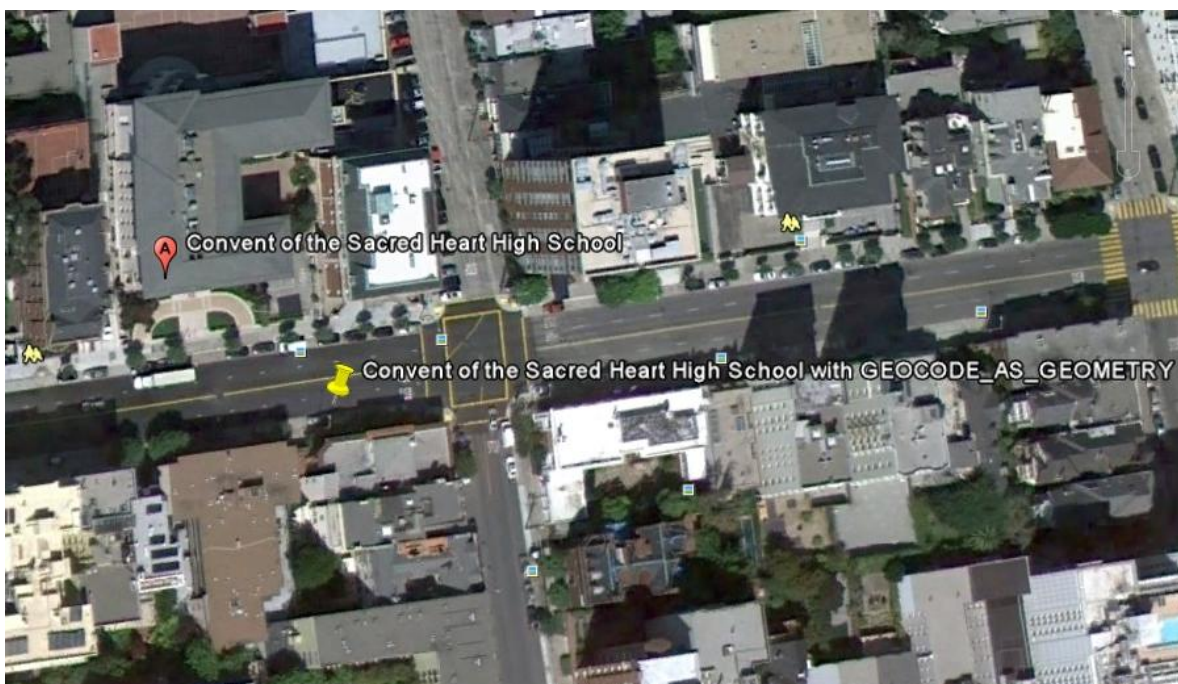


Figure 8-5: School position geocoded with `GEOCODE_AS_GEOMETRY` and presented as KML file with Google Earth and real school position (Map scale 1:95 meters)

Because we working with location information from the City of San Francisco, United States, we used United States National Map Accuracy Standards (US NMAS), to perform horizontal positional accuracy check (U.S. Bureau of the Budget, 1947) and to define acceptable positional accuracy for geocoded results. US NMAS is not explained or researched in this thesis, because it covers very wide spectrum and it requires huge amount of time, which are not available at the moment. According to this standard, “acceptable positional accuracy” for the maps on publication scales larger than 1:20,000 is: Not more than 10 percent of the points tested shall be in error by more than 1/30 inch, measured on the publication scale. Thus, to determine the minimum standards for horizontal accuracy for our data, in actual ground meters, and because our map scale is 1:95 (Figure 8-5) - larger than 1:20.000, the following calculation must be performed (United States Geological Survey, 2009):

$$.03333 \times \text{scale} \times 2.54 / 100 = \text{ground meters}$$

In this calculation, “scale” represents map scale we use for presentation and visualisation of geocoded results, while “ground meters” represent maximum allowed deviation of the geocoded result from its “true location” on the earth.

According to this calculation and already known map scale (1:95 meters – Figure 8-5), we perform:

$$.03333 \times 95 \times 2.54 / 100 = 0,0803529 \text{ m} = 8 \text{ cm}$$

After calculation of a minimum standard for horizontal accuracy (acceptable positional accuracy minimum) in actual ground meters, which is 8 cm, measurement of the distance between result derived with GEOCODE_AS_GEOMETRY for the school “Convent of the Sacred Heart High School” and “true position” of the object presented in Google Earth is executed

(Figure 8-6). The result of the measurement distance between two observed objects is 36.83 meters. However, the result of the measurement can deviate regarding of the position selected as a measurement starting point on Google Earth. As a starting position of the measurement, I used default location provided from Google Earth of the mentioned school. According to the measurement information and minimum standard for horizontal accuracy derived in formula above, which is 0,0803529 meters, we can conclude, that our result does not comply with UN NMAS.

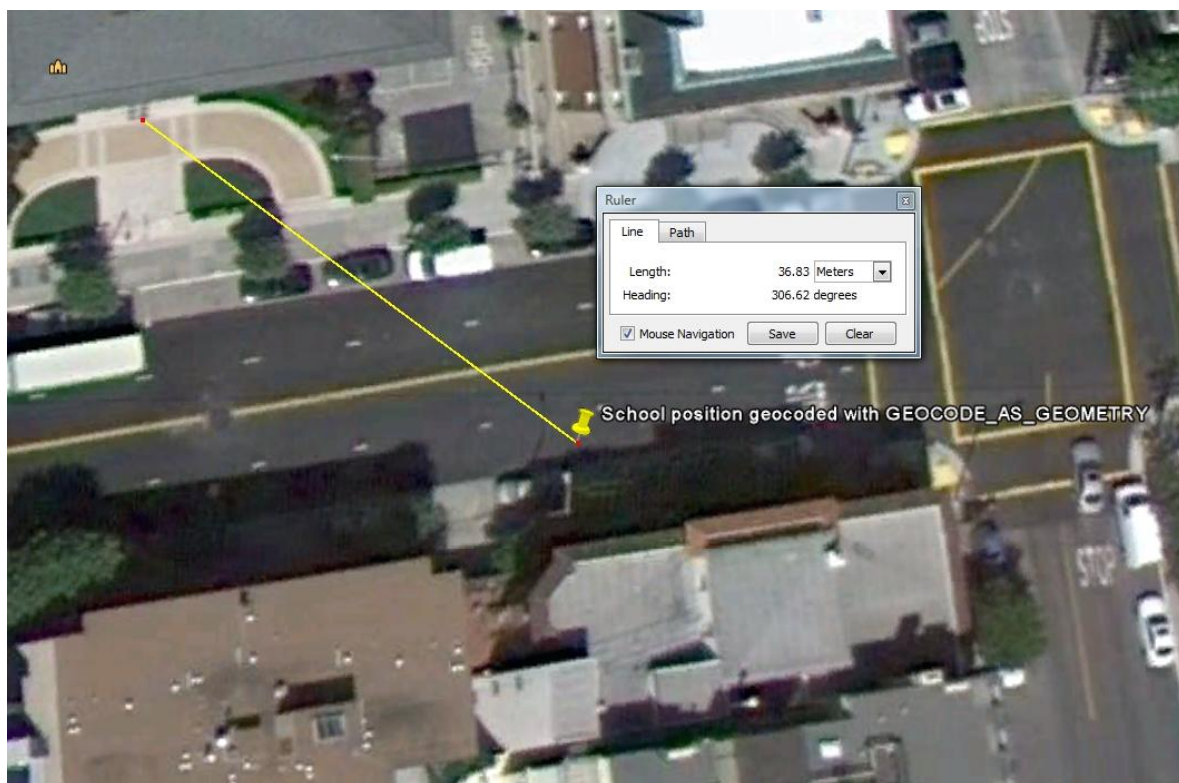


Figure 8-6: Screenshot of distance measurement in Google Earth

Other results as well, show variable amount of deviation from the “true location” (see definition on Page 72) regarding Google Earth. The difference in deviation depends on the type of the street and percent of regularity of buildings distribution within certain street. As shown in Table 8-1, “The Urban School of San Francisco” which is located in Page Street in the City of San Francisco, has the very best accuracy regarding other schools. This is because this street has quite evenly

distributed buildings (Figure 8-7), while other streets show greater level of irregularity of buildings along them. With every result of geocoded school, deviation was larger than allowed by US NMAS. All deviations researched and evaluated here are presented in Table 8-1.



Figure 8-7: Distribution of the buildings along Page St. (yellow line) in city of San Francisco and The Urban School of San Francisco

Table 8-1: Deviations of the schools produced with GEOCODE_AS_GEOMETRY from their true location within Google Earth

School name	Deviation in meters
Convent of the Sacred Heart High School	36,83
Galileo Academy of Science and Technology	18,77
Saints Peter and Paul School	17,62
Sutro Elementary School	28,95
Bay School of San Francisco	27,63
Creative Arts Charter School k-8	26,35
The Urban School of San Francisco	4,93

IHS - French American International School	16,31
Mission High School	25,86
Archbishop Riordan	45,65

According to information and facts provided above, and even using simple visual inspection of the map, we can conclude that GEOCODE_AS_GEOMETRY cannot be effectively used where maps at large scales are produced. Additionally, it cannot be used where good address positional accuracy of the result that can be complied with US NMAS, has importance. Such situations are, for example: accurate planning and facilities positions research, and detailed engineering-based applications. To make our result set complies with US NMAS, we need to execute manual data update (with for example ArcGIS or GeoMedia), which in the case of large amount of data can cost a lot of money and huge amount of time. However, the results of some case studies indicated that manual geocode correction using a web-based interactive approach is a feasible and cost effective method for improving the quality of geocoded resulting set. The level of effort required varies depending on the type of data geocoded (Goldberg, et al., 2008). To perform this, we also need good reference data.

However, data derived during geocoding with GEOCODE_AS_GEOMETRY function can be useful in wide spectrum of application that does not need to comply with US NMAS. These situations, for example, are:

- Spatial analysis of clients and areas they come from
- Optimization of delivery based on the number and distribution of clients
- Map presentation of the urban areas with small and large number of clients

Also, data derived with `GEOCODE_AS_GEOMETRY` function can be used in map presentation with small scale, such as 1:100.000 and smaller. Point location in this case will be accurate and will be complied with US NMAS.

The ideal environment regarding input data and observed spatial location extent for this function does not exist. However, results can be accurate by US NMAS standard, if we deal with geocoding of the points of interest in the street where houses are properly arranged through out that street. However, even in this case, results would not be 100 percent accurate.

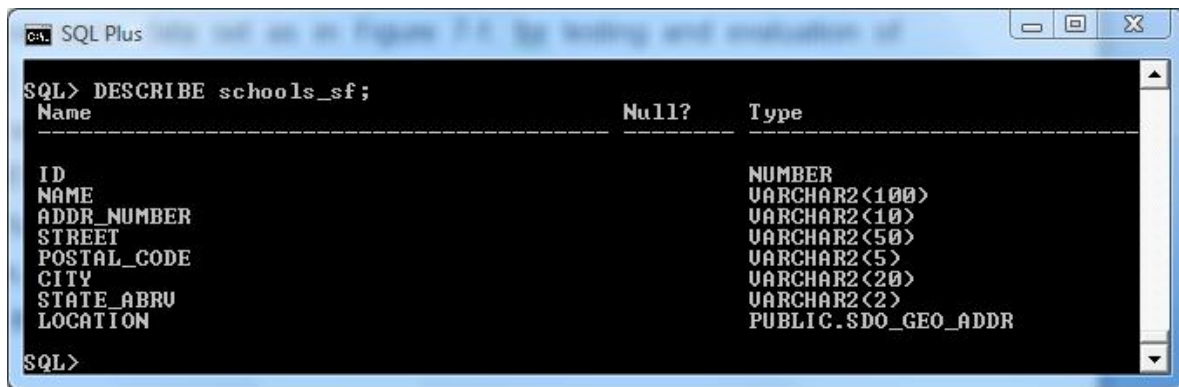
Because `GEOCODE_AS_GEOMETRY` function does not return any additional statistical data, we are unable to provide information for example for standard deviation or mean of the errors for our result set.

8.2 Geocoding with GEOCODE

As we already explained in Chapter 7 of this thesis, this is the very main function of the Oracle geocoder. It returns huge amount of information about a geocoded address and most important, as presented in Table 7-3, it can accept unstructured addresses as an input data set for purpose of geocoding.

8.2.1 Implementation

We will use the same data set as in Figure 8-2 (`gc.dmp`) for testing and evaluation of the `GEOCODE` function. Additionally, to implement this function within Oracle geocoder on our sample data set, we had to change the type of `location` column from `SDO_GEOMETRY` data type to `SDO_GEO_ADDR`. The structure of the table is presented in Figure 8-8 derived with SQLPlus. As we can see, now, instead of `location` column with data type `SDO_GEOMETRY`, we have `location` column which is a type of `SDO_GEO_ADDR`.



SQL> DESCRIBE schools_sf;

Name	Null?	Type
ID		NUMBER
NAME		VARCHAR2(100)
ADDR_NUMBER		VARCHAR2(10)
STREET		VARCHAR2(50)
POSTAL_CODE		VARCHAR2(5)
CITY		VARCHAR2(20)
STATE_ABRV		VARCHAR2(2)
LOCATION		PUBLIC.SDO_GEO_ADDR

SQL>

Figure 8-8: schools_sf table prepared for GEOCODE function

This is more advanced function of geocoder than GEOCODE_AS_GEOMTERY and besides coordinates, SRID and type of the feature it returns fully formatted addresses, and codes that tell us about the “quality” of our geocoding process. Because, we already saved data about addresses and location names in database (Figure 8-2) we can execute GEOCODE function. Code for execution of this data over already existing data in our schools_sf table is presented Listing 8-4.

```
UPDATE schools_sf
SET LOCATION =
(
SELECT SDO_GCDR.GEOCODE
(
'SPATIAL',
SDO_KEYWORDARRAY(addr_number || ' ' || street, city || ', '
|| state_abrv),
'US',
'DEFAULT'
)
FROM DUAL);
```

Listing 8-4: Updating location column of schools_sf table with GEOCODE function

8.2.2 Result

After execution of SELECT statement over schools_sf table for location column, we see spatially geocoded results. Thus, we know that we hold

geographical coordinates for every school in “location” column. We can present schools of San Francisco on the map or spatially analyse them. In addition to the result set for `location` column in Listing 8-3 (page 69), now we have additional information for every school in our `location` column (Listing 8-5). However, if we compare geographical coordinates for every school that are produced with either `GEOCODE_AS_GEOMETRY` or `GEOCODE` function, we will notice that they are exactly same. Thus, map presentation in both cases is the same.

Result of `SELECT id, name, location from schools_sf` query is presented in Listing 8-5, visually enhanced with SQLPlus, because of better and finer overview of result set.

```
SQL> select id, name, location as location from schools_sf;
```

ID	NAME	LOCATION
3	Convent of the Sacred Heart High School	SDO_GEO_ADDR(0, SDO_KEYWORDARRAY(), NULL, 'BROADWAY ST', NULL, NULL, 'SAN FRANCISCO', NULL, 'CA', 'US', '94115', NULL, '94115', NULL, '2222', 'BROADWAY', 'ST', 'F', 'F', NULL, NULL, 'L', .775510204, 23598822, '????#ENUT?B281CP?', 1, 'DEFAULT', -122.4336, 37.7944029, '???10101010??004?')
4	Galileo Academy of Science & Technology	SDO_GEO_ADDR(0, SDO_KEYWORDARRAY(), NULL, 'FRANCISCO ST', NULL, NULL, 'SAN FRANCISCO', NULL, 'CA', 'US', '94109', NULL, '94109', NULL, '1150', 'FRANCISCO', 'ST', 'F', 'F', NULL, NULL, 'L', .489795918, 23604420, '????#ENUT?B281CP?', 1, 'DEFAULT', -122.42402, 37.803308, '???10101010??004?')
5	Sts Peter & Paul School	SDO_GEO_ADDR(0, SDO_KEYWORDARRAY(), NULL, 'FILBERT ST', NULL, NULL, 'SAN FRANCISCO', NULL, 'CA', 'US', '94133', NULL, '94133', NULL, '666', 'FILBERT', 'ST', 'F', 'F', NULL, NULL, 'L', .326530612, 23603974, '????#ENUT?B281CP?', 1, 'DEFAULT', -122.41034, 37.8012618, '???10101010??004?')
1	Sutro Elementary School	SDO_GEO_ADDR(0, SDO_KEYWORDARRAY(), NULL, '12TH AVE', NULL, NULL, 'SAN FRANCISCO', NULL, 'CA', 'US', '94118', NULL, '94118', NULL, '235', '12TH', 'AVE', 'F', 'F', NULL, NULL, 'L', .653061224, 23594256, '????#ENUT?B281CP?', 1, 'DEFAULT', -122.470088, 37.7838678, '???10101010??004?')

2 Bay School of San Francisco	SDO_GEO_ADDR(0, SDO_KEYWORDARRAY(), NULL, 'KEYES AVE', NULL, NULL, 'SAN FRANCISCO', NULL, 'CA', 'US', '94129', NULL, '94129', NULL, '35', 'KEYES', 'AVE', 'F', 'F', NULL, NULL, 'L', .055555556, 23622600, '????#ENUT?B281CP?', 1, 'DEFAULT', -122.45634, 37.8002211, '???10101010??004?')
6 Creative Arts Charter School k-8	SDO_GEO_ADDR(0, SDO_KEYWORDARRAY(), NULL, 'TURK ST', NULL, NULL, 'SAN FRANCISCO', NULL, 'CA', 'US', '94115', NULL, '94115', NULL, '1601', 'TURK', 'ST', 'F', 'F', NULL, NULL, 'R', 1, 23619099, '????#ENUT?B281CP?', 1, 'DEFAULT', -122.43538, 37.78001, '???10101010??004?')
7 The Urban School of San Francisco	SDO_GEO_ADDR(0, SDO_KEYWORDARRAY(), NULL, 'PAGE ST', NULL, NULL, 'SAN FRANCISCO', NULL, 'CA', 'US', '94117', NULL, '94117', NULL, '1563', 'PAGE', 'ST', 'F', 'F', NULL, NULL, 'R', .367346939, 23613431, '????#ENUT?B281CP?', 1, 'DEFAULT', -122.44642, 37.7709808, '???10101010??004?')
8 IHS - French American International School	SDO_GEO_ADDR(0, SDO_KEYWORDARRAY(), NULL, 'OAK ST', NULL, NULL, 'SAN FRANCISCO', NULL, 'CA', 'US', '94102', NULL, '94102', NULL, '150', 'OAK', 'ST', 'F', 'F', NULL, NULL, 'L', .489795918, 23612634, '????#ENUT?B281CP?', 1, 'DEFAULT', -122.42175, 37.7750829, '???10101010??004?')
9 Mission High School	SDO_GEO_ADDR(0, SDO_KEYWORDARRAY(), NULL, '18TH ST', NULL, NULL, 'SAN FRANCISCO', NULL, 'CA', 'US', '94114', NULL, '94114', NULL, '3750', '18TH', 'ST', 'F', 'F', NULL, NULL, 'L', .489795918, 23594644, '????#ENUT?B281CP?', 1, 'DEFAULT', -122.42728, 37.7612437, '???10101010??004?')
10 Archbishop Riordan	SDO_GEO_ADDR(0, SDO_KEYWORDARRAY(), NULL, 'PHELAN AVE', NULL, NULL, 'SAN FRANCISCO', NULL, 'CA', 'US', '94112', NULL, '94112', NULL, '175', 'PHELAN', 'AVE', 'F', 'F', NULL, NULL, 'L', .368421053, 23748110, '????#ENUT?B281CP?', 1, 'DEFAULT', -122.45239, 37.7273505, '???10101010??004?')

10 rows selected.

Listing 8-5: SELECT statement and results derived with GEOCODE function

In addition to exactly the same produced coordinate results for every school, we can see, there is significantly large amount of information provided for `location` column then in example in listing 8-3. It is obvious that a structure of `location` column in Figure 8-5 contains very detailed results of geocoding function from Listing 8-4. It is quite rich if we compare it with result set of geocoding with `GEOCODE_AS_GEOMETRY`.

To explain results and better understand the result set derived with `GEOCODE` function, I created parallel presentation of `SDO_GEO_ADDR` attributes and location information for one school with `id` 3 from Listing 8-5. Table 8-2 shows this presentation. In the left column of the Table 8-2, we see `SDO_GEO_ADDR` attributes (one per line), while in right column we see corresponding geocoded value for these attributes for school with `id` 3 (one per line as well). Explanation of every `SDO_GEO_ADDR` attribute is provided in table in Appendix III.

Table 8-2: `SDO_GEO_ADDR` attributes and their values for `GEOCODE` function result

SDO_GEO_ADDR attributes	School with ID 3 from Listing 8-5
<code>Id</code>	NULL
<code>AddressLines</code>	<code>SDO_KEYWORDARRAY()</code>
<code>PlaceName</code>	NULL
<code>StreetName</code>	BROADWAY ST
<code>IntersectStreet</code>	NULL
<code>SecUnit</code>	NULL
<code>Region</code>	San Francisco
<code>Municipality</code>	NULL
<code>Country</code>	US
<code>PostalCode</code>	94115
<code>PostalAddOnCode</code>	NULL
<code>FullPostalCode</code>	94115
<code>POBox</code>	NULL
<code>HouseNumber</code>	2222

BaseName	BROADWAY
StreetType	ST
StreetTypeBefore	F
StreetTypeAttached	F
StreetPrefix	NULL
StreetSuffix	NULL
Side	L
Percent	0.775510204
EdgeID	23598822
ErrorMessage	????#ENUT?B281CP?
MatchCode	1
MatchMode	DEFAULT
Longitude	-122.4336
Latitude	37.7944029
MatchVector	???10101010??004?

8.2.3 Analysis of the result

Geocoded coordinates with both functions (GEOCODE_AS_GEOMETRY and GEOCODE) are exactly the same. However, additional information about geocoded results are not. As we can see in Table 8-2 or Listing 8-5 the result set for “location” column produced with the function GEOCODE has more information than result set produced with GEOCODE_AS_GEOMETRY function (Listing 8-3). Result set produced with GEOCODE function gives us information about matching value and percentage for results deviation and many more. It provides us with correctly formatted addresses, which in addition allows for further querying and analysis. However, we do not hold information about a type of geometry and SRID. We can derive this information if we query our result set with MapViewer for geometry type and if we see SRID in our reference data. Because of additional information like percentage of deviation and matching code, we are able to define accuracy value of our data and calculate statistical operations like standard

deviation or mean. Also, it gives us indication on which side of the street our point of interest is.

Results produced with GEOCODE function have not acceptable positional accuracy if we intend to use our data with large map scales. If we present them on the map, or query for coordinates for specific school, the result would be the same. However, this function can be seen as useful if we need to correct our input data and get additional information about investigated address.

Even GEOCODE produces more information as a result for “location” and its position, then GEOCODE_AS_GEOMETRY, we cannot use it when we need data with good positional accuracy that comply with US NMAS. We can do so, if we manually correct all of our locations, with par example ArcGIS or Geomedia tools. If we decide to do so, we have to be aware of the fact about size of our data set. But, the same as with results produced with GEOCODE_AS_GEOMETRY, it can be useful for other spatial analysis where good positional accuracy of the data is not required, and discovering information about customers, clients, their positions etc.

The map derived with this data set, would be the same as with one derived with results of GEOCODE_AS_GEOMETRY function (Figures: 8-3 and 8-4), but in addition we can produce much more information, which would allow us to better understand position of point of interest and its correlation with real object.

8.3 Geocoding with GEOCODE_ADDR

This function is identical as GEOCODE, but it can only accept SDO_GEO_ADDR object as an input, instead SDO_KEYWORDARRAY. Additionally when working with this function, we have to use structured addresses as an input data set.

8.3.1 Implementation

When working with GEOCODE_ADDR function, we have to fill in all arguments to the SDO_GEO_ADDR constructor (NULL can be used as well, if we don't have value for some arguments). Otherwise, function will not work. We will work on the same data sample as before. In addition we have to delete all values in `location` column in our `schools_sf` table. After I have deleted all values from `location` column, I executed `UPDATE` statement to automatically populate `location` column from already existing data in `schools_sf` table. The `UPDATE` query can be seen in Listing 8-6.

```
update spatial.schools_sf
set location =
(
SELECT SDO_GCDR.GEOCODE_ADDR
(
'spatial',
SDO_GEO_ADDR
(
'US',
'DEFAULT',
ADDR_NUMBER || ' ' || STREET,
CITY,
NULL,
STATE_ABRV,
POSTAL_CODE
)
)
FROM DUAL);
```

Listing 8-6: Updating location column using geocoder function GEOCODE_ADDR

8.3.2 Result and Analysis

The result of the executed query and analysis of produced results is identical as with GEOCODE function (Figure 8-9). To compare geocoded results of both functions, we use columns that provide coordinates in Figure 8-9 and Listing 8-5 for selected school.


```

SQL> select id, name, location AS LOCATION from schools_sf;

```

ID	NAME	LOCATION(ID, ADDRESSLINES, PLACENAME, STREETNAME, INTERSECT
1	Sutro Elementary School	SDO_GEO_ADDR(0, SDO_KEYWORDARRAY(), NULL, '12TH AVE', NULL, NULL, 'SAN FRANCISCO', NULL, 'CA', 'US', '94118', NULL, '94118', NULL, '235', '12TH', 'AVE', 'F', 'F', NULL, NULL, 'L', '653061224, 23594256, '????#ENUT?B281CP?', 1, 'DEFAULT', -122.47088, 37.7838678, '???10101014??000?')
3	Convent of the Sacred Heart High School	SDO_GEO_ADDR(0, SDO_KEYWORDARRAY(), NULL, 'BROADWAY ST', NULL, NULL, 'SAN FRANCISCO', NULL, 'CA', 'US', '94115', NULL, '94115', NULL, '2222', 'BROADWAY', 'ST', 'F', 'F', NULL, NULL, 'L', '775510204, 23598822, '????#ENUT?B281CP?', 1, 'DEFAULT', -122.4336, 37.7944029, '???10101014??000?')
4	Galileo Academy of Science & Technology	SDO_GEO_ADDR(0, SDO_KEYWORDARRAY(), NULL, 'FRANCISCO ST', NULL, NULL, 'SAN FRANCISCO', NULL, 'CA', 'US', '94109', NULL, '94109', NULL, '1150', 'FRANCISCO', 'ST', 'F', 'F', NULL, NULL, 'L', '489795918, 23604420, '????#ENUT?B281CP?', 1, 'DEFAULT', -122.42402, 37.803308, '???10101014??000?')
5	Sts Peter & Paul School	SDO_GEO_ADDR(0, SDO_KEYWORDARRAY(), NULL, 'FILBERT ST', NULL, NULL, 'SAN FRANCISCO', NULL, 'CA', 'US', '94133', NULL, '94133', NULL, '666', 'FILBERT', 'ST', 'F', 'F', NULL, NULL, 'L', '326530612, 23603974, '????#ENUT?B281CP?', 1, 'DEFAULT', -122.41034, 37.8012618, '???10101014??000?')
2	Bay School of San Francisco	SDO_GEO_ADDR(0, SDO_KEYWORDARRAY(), NULL, 'KEYES AVE', NULL, NULL, 'SAN FRANCISCO', NULL, 'CA', 'US', '94129', NULL, '94129', NULL, '35', 'KEYES', 'AVE', 'F', 'F', NULL, NULL, 'L', '055555556, 23622600, '????#ENUT?B281CP?', 1, 'DEFAULT', -122.45634, 37.8002211, '???10101014??000?')
6	Creative Arts Charter School	SDO_GEO_ADDR(0, SDO_KEYWORDARRAY(), NULL, 'TURK ST', NULL, NULL, 'SAN FRANCISCO', NULL, 'CA', 'US', '94115', NULL, '94115', NULL, '1601', 'TURK', 'ST', 'F', 'F', NULL, NULL, 'R', '1, 23619099, '????#ENUT?B281CP?', 1, 'DEFAULT', -122.43538, 37.78001, '???10101014??000?')
7	The Urban School of San Francisco	SDO_GEO_ADDR(0, SDO_KEYWORDARRAY(), NULL, 'PAGE ST', NULL, NULL, 'SAN FRANCISCO', NULL, 'CA', 'US', '94117', NULL, '94117', NULL, '1563', 'PAGE', 'ST', 'F', 'F', NULL, NULL, 'R', '367346939, 23613431, '????#ENUT?B281CP?', 1, 'DEFAULT', -122.44642, 37.7709808, '???10101014??000?')
8	IHS - French American International School	SDO_GEO_ADDR(0, SDO_KEYWORDARRAY(), NULL, 'OAK ST', NULL, NULL, 'SAN FRANCISCO', NULL, 'CA', 'US', '94102', NULL, '94102', NULL, '150', 'OAK', 'ST', 'F', 'F', NULL, NULL, 'L', '489795918, 23612634, '????#ENUT?B281CP?', 1, 'DEFAULT', -122.42175, 37.7750829, '???10101014??000?')
9	Mission High School	SDO_GEO_ADDR(0, SDO_KEYWORDARRAY(), NULL, '18TH ST', NULL, NULL, 'SAN FRANCISCO', NULL, 'CA', 'US', '94114', NULL, '94114', NULL, '3750', '18TH', 'ST', 'F', 'F', NULL, NULL, 'L', '489795918, 23594644, '????#ENUT?B281CP?', 1, 'DEFAULT', -122.42728, 37.7612437, '???10101014??000?')
10	Archbishop Rioran	SDO_GEO_ADDR(0, SDO_KEYWORDARRAY(), NULL, 'PHELAN AVE', NULL, NULL, 'SAN FRANCISCO', NULL, 'CA', 'US', '94112', NULL, '94112', NULL, '175', 'PHELAN', 'AVE', 'F', 'F', NULL, NULL, 'L', '368421053, 23748110, '????#ENUT?B281CP?', 1, 'DEFAULT', -122.45239, 37.7273505, '???10101014??000?')

```

10 rows selected.
SQL>

```

Figure 8-9: SELECT statement and results derived with GEOCODE_ADDR function

9 Summary

9.1 Final word

The broad objective of this thesis was to examine the functionality, usability, accuracy and application of automated geocoding process using Oracle geocoder functionality, and its functions that return only one matched location regarding reference data. It is shown that this process is feasible and if criteria are satisfied - successful. Using theoretical studies of available literature in this work, it is shown, that automated geocoding process of addresses can help companies, individuals, and institutions in spatial analysis and map visualisation of certain location of the customers, clients, partners and many other entities. However, accuracy assessment of geocoded results showed that data with inaccurate geographical coordinates that does not comply with national mapping accuracy standards would be produced for large map scales. In addition, data with acceptable accuracy for small map scales can be produced using geocoding functions of Oracle Spatial. Method for simple accuracy assessment of geocoded results using Google Earth as a reference tool is presented as well.

9.2 Conclusions

This thesis answered to several questions, and conclusions that can be derived from key findings are presented:

Is it possible to convert addresses to geographical coordinates with good and acceptable positional accuracy, according to US NMAS specification, using Oracle Spatial geocoder?

Accuracy assessment of produced result showed that geocoding resulting set will always be inaccurate when speaking about geographical coordinates. However,

some level of inaccuracy, especially with small map scales that complies with US NMAS can be accepted. It is possible to produce geographical coordinates that have good and acceptable positional accuracy for the purpose of spatial analysis, but only for the purpose, for example, of small map scale presentation. To use resulting set for the purpose of large map scale presentation, geocoded result must be corrected which in most cases requires additional resources, such as money and people.

How to convert available addresses to geographical coordinates using Oracle Spatial?

In chapters 7 and 8, we explained the process of converting addresses and producing additional information from already existing data in Oracle 11g, with the help of geocoding functions that return one matched location in correlation with referenced data.

Is it possible to perform accuracy assessment of geocoded results with Google Earth as a reference tool for accuracy validation?

It is demonstrated that accuracy assessment of geocoded result can be performed using Google Earth as a reference tool. However, validity and quality of this method is not evaluated.

What can we expect from geocoded data in the terms of validity of resulting set, except geographical coordinates? In other words, how valid are other geocoded data beside geographic coordinates, such as street name, city name, etc?

All other data in geocoding resulting set showed correct state. During the process of geocoding, other data, such as the street name, were exposed to the process of

correction (if needed). Thus, resulting set provides us with corrected and structured geographical information, such as street name, name of the city, etc.

Can geocoded data be presented on the maps and integrated into web mapping services as Google Maps, Yahoo Maps, etc?

It is demonstrated, through the use of Google Earth in this master thesis it is feasible to integrate and present geocoded data on the internet mapping systems. However, additional work is required (such as export to KML file to use with Google Earth) to be able to use them.

9.3 Future work

There are additional functions in Oracle Spatial, such as GEOCODE_ALL and GEOCODE_ADDR_ALL which need to be evaluated and investigated regarding the process of geocoding. Thus, the results of this work can be used as a base for further research of the geocoder functions with multiple matching results. As demonstrated, we used Google Earth as a reference tool for accuracy assessment. However, validity of accuracy assessment with Google Earth and research regarding this is not provided in this work. Thus, this work gives decent base for research regarding validity of accuracy assessment of geocoded results with Google Earth. Methods for geocoding presented in this thesis are based mostly on several months work, so, there is quite larger outlook for the further research regarding theoretical background and case studies. Amount of data used as a sample data set for the purpose of evaluation and investigation of geocoding function is reasonably small, thus there is a large field of possible investigation regarding larger and more complex data sets.

References

Arctur, David, and Michael Zeiler. *Designing geodatabases: case studies in GIS data modeling*. Redlands, California: ESRI, Inc., 2004.

Ashdown, Lance, and Tom Kyte. *Oracle® Database Concepts 11g Release 2 (11.2)*. California, United States: Oracle Corporation, 2009.

Brinker, Russell Charles, and Roy Minnick. *The surveying handbook*. New York, United States: Springer, 1995.

Codd, E. F. *Derivability, Redundancy, and Consistency of Relations Stored in Large Data Banks*. San Jose, California, United States: IBM Research Report, 1969.

—. *The relational model for database management: Version 2*. Reading, Massachusetts, United States: Addison-Wesley (Original from University of California), 1990.

Davison, Robert M., Maris G. Martinsons, and Ned Kock. *Principles of Canonical Action Research*. *Information Systems Journal*, Volume 14, Number 1, 65. United States, 2004.

Goldberg, Daniel W, John P Wilson, Craig A Knoblock, Beate Ritz, and Myles G Cockburn. *An effective and efficient approach for manually improving geocoded data*, *International Journal of Health Geographics* 2008, 7:60. Los Angeles, CA, USA: University of Southern California, 2008.

ISRD-Group. *Database - an Introduction in Introduction to Database Management Systems*. Columbus, United States: The McGraw-Hill Companies, 2006.

- Joseph, Joshy, and Craig Fellenstein. *Grid computing, IBM press series-Information management: On demand computing books*. New Jersey, United States: Prentice Hall PTR, 2004.
- Kogent Solutions Inc. *Oracle 11g Administration in Simple Steps*. Delhi, India: Dreamtech Press, 2008.
- Kothuri, Ravi, Albert Godfrind, and Euro Beinat. *Pro Oracle Spatial for Oracle Database 11g (First Edition)*. California, United States: Apress, 2007.
- Krishna, S. *Introduction to database and knowledge-base systems, Volume 28 of Series in computer science*. Singapore, Singapore: World Scientific, 1992.
- Kutz, Myer. *Handbook of transportation engineering*. New York City, United States: McGraw-Hill Professional, 2004.
- Melnick, Alan L. *Introduction to geographic information systems in public health in Public Health Series*. Sudbury, Massachusetts, United States: Jones & Bartlett Learning, 2002.
- Miller, Frederic P, Agnes F Vandome, and John McBrewster. *Geocoding*. Mauritius: VDM Publishing House Ltd., 2010.
- Moore, Sheila. *Oracle® Database PL/SQL Language Reference 11g Release 1 (11.1)*. Redwood Shores, California, United States: Oracle Corporation, 2007.
- Murray, Chuck. *Oracle® Spatial User's Guide and Reference 10g Release 1 (10.1)*. Redwood Shores, California, United States: Oracle Corporation, 2003.
- Narang, Rajesh. *Database Management Systems*. New Delhi, India: PHI Learning Pvt. Ltd., 2006.
- Oppel, Andy, and Robert Sheldon. *SQL: a beginner's guide, Beginner's Guide*. Columbus, United States: McGraw-Hill Professional, 2008.

Pick, James B. *Geo-business: GIS in the digital organization*. Hoboken, New Jersey, United States: John Wiley and Sons, 2008.

Pribyl, Bill, and Steven Feuerstein. *Learning Oracle PL/SQL in Learning Series Oracle Development Languages*. United States: O'Reilly Media, Inc., 2002.

PSMA Australia. *Transport & Topography, Version 1.7*. Australia: PSMA Australia Limited, May 2010.

Rigaux, Philippe, Michel O. Scholl, and Agnes Voisard. *Introduction to spatial databases: with application to GIS*. United States: Morgan Kaufmann, 2002.

Rob, Peter, and Carlos Coronel. *Database systems: Design, Implementation, and Management*. Andover, United Kingdom: Cengage Learning, 2009.

Schofield, Jack, and Emma Brockes. *The Guardian*. 28 April 2010.
<http://www.guardian.co.uk/g2/story/0,3604,215072,00.html> (accessed November 6, 2010).

Strohm, Richard. *Oracle® Database Concepts 11g Release 1 (11.1)*. California, United States: Oracle Corporation, 2008.

Sumathi, S., and S. Esakkirajan. *Fundamentals of Relational Database Management Systems, Volume 47 of Studies in computational intelligence*. Berlin, Germany: Springer- Verlag Belin Heidelberg , 2007.

Throll, Markus, and Oliver Bartosch. *Einstieg in SQL*. Bonn, Germany: Galileo Press, 2010.

U.S. Bureau of the Budget. *United States National Map Accuracy Standards*. U.S. Bureau of the Budget, 1947.

United States Geological Survey. *Texas commision on environmental quality*. 24 September 2009. <http://www.tceq.state.tx.us/gis/natmap.html> (accessed January 05, 2011).

Wikipedia - Free Encyclopedia. *Wikipedia*. 9 November 2010.

<http://en.wikipedia.org/wiki/Geocoding> (accessed November 11th, 2010).

Wikipedia. *Wikipedia, the free encyclopedia*. 03 01 2011.

http://en.wikipedia.org/wiki/Google_Earth#Resolution_and_accuracy (accessed January 4th, 2011).

Appendix I

List of addresses of schools of San Francisco without geographical coordinates:

Sutro Elementary School

235 12th Avenue
San Francisco, CA

Bay School of San Francisco

35 Keyes Avenue
San Francisco, CA

Convent of the Sacred Heart High School

2222 Broadway Street
San Francisco, CA

Galileo Academy of Science & Technology

1150 Francisco Street
San Francisco, CA

Sts Peter & Paul School

666 Filbert Street
San Francisco, CA

Creative Arts Charter School k-8

1601 Turk St
San Francisco, CA

The Urban School of San Francisco

1563 Page Street
San Francisco, CA

International High School - French American International School

150 Oak Street
San Francisco, CA

Mission High School

3750 18th Street
San Francisco, CA

Archbishop Riordan

175 Phelan Avenue

San Francisco, CA

Appendix II

SQL Statement listing for data insertion of San Francisco schools table.

```
INSERT INTO schools_sf (id, name, addr_number, street,
postal_code, city, state_abrv)
VALUES
(1, 'Sutro Elementary School', '235', '12th Avenue', '94118',
'San Francisco', 'CA');

INSERT INTO schools_sf (id, name, addr_number, street,
postal_code, city, state_abrv)
VALUES
(2, 'Bay School of San Francisco', '35', 'Keyes Avenue',
'94129', 'San Francisco', 'CA');

INSERT INTO schools_sf (id, name, addr_number, street,
postal_code, city, state_abrv)
VALUES
(3, 'Convent of the Sacred Heart High School', '2222',
'Broadway Street', '94115', 'San Francisco', 'CA');

INSERT INTO schools_sf (id, name, addr_number, street,
postal_code, city, state_abrv)
VALUES
(4, 'Galileo Academy of Science & Technology', '1150',
'Francisco Street', '94109', 'San Francisco', 'CA');

INSERT INTO schools_sf (id, name, addr_number, street,
postal_code, city, state_abrv)
VALUES
(5, 'Sts Peter & Paul School', '666', 'Filbert Street',
'94133', 'San Francisco', 'CA');

INSERT INTO schools_sf (id, name, addr_number, street,
postal_code, city, state_abrv)
VALUES
(6, 'Creative Arts Charter School k-8', '1601', 'Turk St',
'94115', 'San Francisco', 'CA');

INSERT INTO schools_sf (id, name, addr_number, street,
postal_code, city, state_abrv)
VALUES
(7, 'The Urban School of San Francisco', '1563', 'Page
Street', '94117', 'San Francisco', 'CA');
```

```
INSERT INTO schools_sf (id, name, addr_number, street,  
postal_code, city, state_abrv)  
VALUES  
(8, 'IHS - French American International School', '150', 'Oak  
Street', '94102', 'San Francisco', 'CA');  
  
INSERT INTO schools_sf (id, name, addr_number, street,  
postal_code, city, state_abrv)  
VALUES  
(9, 'Mission High School', '3750', '18th Street', '94114',  
'San Francisco', 'CA');  
  
INSERT INTO schools_sf (id, name, addr_number, street,  
postal_code, city, state_abrv)  
VALUES  
(10, 'Archbishop Riordan', '175', 'Phelan Avenue', '94112',  
'San Francisco', 'CA');
```

Appendix III

SDO_GEO_ADDR Type Attributes

Attribute	Data Type	Description
Id	NUMBER	(Not used.)
AddressLines	SDO_KEYWORDARRAY	Address lines.
PlaceName	VARCHAR2(200)	Point of interest (POI) name. Example: CALIFORNIA PACIFIC MEDICAL CTR
StreetName	VARCHAR2(200)	Street name, including street type. Example: MAIN ST
IntersectStreet	VARCHAR2(200)	Intersecting street.
SecUnit	VARCHAR2(200)	Secondary unit, such as an apartment number or building number.
Settlement	VARCHAR2(200)	Lowest-level administrative area to which the address belongs
Municipality	VARCHAR2(200)	Administrative area above settlement.
Region	VARCHAR2(200)	Administrative area above municipality (if applicable), or above settlement if municipality does not apply.
Country	VARCHAR2(100)	Country name or ISO country code.
PostalCode	VARCHAR2(20)	Postal code (optional if administrative area information is provided). In the United States, the postal code is the 5-digit ZIP code.
PostalAddOnCode	VARCHAR2(20)	String appended to the postal code. In the United States, the postal add-on code is typically the last four numbers of a 9-digit ZIP code specified in "5-4" format.
FullPostalCode	VARCHAR2(20)	Full postal code, including the postal code and postal add-on code.
POBox	VARCHAR2(100)	Post Office box number.
HouseNumber	VARCHAR2(100)	House or building number. Example: 123 in 123 MAIN ST
BaseName	VARCHAR2(200)	Base name of the street. Example: MAIN in 123 MAIN ST
StreetType	VARCHAR2(20)	Type of the street. Example: ST in 123 MAIN ST

Attribute	Data Type	Description
StreetTypeBefore	VARCHAR2(1)	(Not used.)
StreetTypeAttached	VARCHAR2(1)	(Not used.)
StreetPrefix	VARCHAR2(20)	Prefix for the street. Example: S in 123 S MAIN ST
StreetSuffix	VARCHAR2(20)	Suffix for the street. Example: NE in 123 MAIN ST NE
Side	VARCHAR2(1)	Side of the street (L for left or R for right) that the house is on when you are traveling along the road segment following its orientation (that is, from its start node toward its end node). The house numbers may be increasing or decreasing.
Percent	NUMBER	Number from 0 to 1 (multiply by 100 to get a percentage value) indicating how far along the street you are when traveling following the road segment orientation.
EdgeID	NUMBER	Edge ID of the road segment.
ErrorMessage	VARCHAR2(20)	Error messageNote: You are encouraged to use the MatchVector attribute instead of the ErrorMessage attribute.
MatchCode	NUMBER	Match code.
MatchMode	VARCHAR2(30)	Match mode.
Longitude	NUMBER	Longitude coordinate value.
Latitude	NUMBER	Latitude coordinate value.
MatchVector	VARCHAR2(20)	A string that indicates how each address attribute has been matched against the data used for geocoding

Appendix IV

Schools of San Francisco as exported KML file (contains name and location columns)

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.google.com/earth/kml/2">

<Document>
<name>Schools of San Francisco</name>

<Placemark>
<name>Sutro Elementary School</name>
  <Point>
    <coordinates>-
122.470877959184,37.783867755102</coordinates>
  </Point>
</Placemark>

<Placemark>
<name>Convent of the Sacred Heart High School</name>
  <Point>
    <coordinates>-
122.433596938776,37.7944028571429</coordinates>
  </Point>
</Placemark>

<Placemark>
<name>Galileo Academy of Science and Technology</name>
  <Point>
    <coordinates>-
122.424021632653,37.8033079591837</coordinates>
  </Point>
</Placemark>

<Placemark>
<name>Saints Peter and Paul School</name>
  <Point>
    <coordinates>-
122.410341428571,37.8012618367347</coordinates>
  </Point>
```

```

</Placemark>

<Placemark>
<name>Bay School of San Francisco</name>
  <Point>
    <coordinates>-
122.456343888889,37.8002211111111</coordinates>
  </Point>
</Placemark>

<Placemark>

<name>Creative Arts Charter School k-8</name>
  <Point>
    <coordinates>-122.43538,37.78001</coordinates>
  </Point>
</Placemark>

<Placemark>
<name>The Urban School of San Francisco</name>
  <Point>
    <coordinates>-
122.446424897959,37.7709808163265</coordinates>
  </Point>
</Placemark>

<Placemark>
<name>IHS - French American International School</name>
  <Point>
    <coordinates>-
122.421746326531,37.7750828571429</coordinates>
  </Point>
</Placemark>

<Placemark>
<name>Mission High School</name>
  <Point>
    <coordinates>-
122.42727755102,37.7612436734694</coordinates>
  </Point>
</Placemark>

```

```
<Placemark>
  <name>Archbishop Riordan</name>
    <Point>
      <coordinates>-
122.452392425021,37.7273505426575</coordinates>
    </Point>
</Placemark>

</Document>
</kml>
```