DESIGN OF NATURAL LANGUAGES TRANSLATORS TO SQL (*STRUCTURED QUERY LANGUAGE*) FORMAT WITH FOCUS ON RETRIVAL CASE INFORMATION

Ihsan Faturohman[1], Galih Hermawan[2]

Teknik Informatika – Universitas Komputer Indonesia
Jl. Dipatiukur No. 112-116, Bandung 40132
E-mail : ihsan.seeking@gmail.com[1], galih.hermawan@email.unikom.ac.id[2]

ABSTRACT

Database is a storage of information, but not everyone can access the information, because it requires a special skill, namely understand the standard format called SQL syntax (Structured Query Language). Not all input commands used to access information are SQL syntax. Often the input command is a natural language. SQL syntax certainly has a different format with the natural language of Indonesia. Therefore we need a tool that can translate natural language into SQL format as a bridge to access information from DBMS. A natural language translator system built using the rule-base method to determine the translation phase, based on the extraction of information from the command line. there are four process blocks, which are database analysis to extract the database information, then preprocessing sentences to idealize the input sentence, after which enter the Translation process phase, which consists of identification command, table, attribute, feature, and query compilation. the last result of the query is extracted to be tested for its success in displaying the data in question. Based on the results of testing the function of the system interface has been running quite well. the output of this system has an accuracy ranging from 72.22% - 83.87% for testing 6 types of queries totaling 18 test sentences.

Keywords : Information Extraction, Rule - Based, Query, NLP, Natural Language.

1. INTRODUCTION

Database is a storage of information, but not everyone can access the information, because it requires a special skill, namely understand the standard format called SQL syntax (Structured Query Language). Syntax SQL is a language used by MySQL DBMS. DBMS (Database Management System) itself is an engine or software used to manage the database. Processing of structures and data on DBMS is also divided into 2 ie DDL (Data Definition Language) and DML (Data Manipulation Language). DDL is a SQL syntax structure that focuses on building a container for information while DML is a SQL syntax structure that focuses on data manipulation, such as, storing, modifying, deleting, and displaying information. Not all input commands used to access information are SQL syntax. Often the input command is a natural language. SQL syntax certainly has a different format with the natural language of Indonesia. Therefore we need a tool that can translate natural language into SQL format as a bridge to access information from DBMS.

This research focuses on the extraction of information in the translation of sentences into the SQL syntax to access information contained in the DBMS. because this study focuses on the retrival case of information then some of the problem limits are applied as follows:

a. The SQL syntax analyzed in this study focuses on Retrival Information ie SELECT queries.
b. SELECT query has 6 types, that is, queries without conditions, queries with conditions, queries Many conditions, queries many tables without conditions, queries many tables many conditions, query order.

The objectives to be achieved from this research are:

a. Provides a system that serves as a natural language translator into SQL format as a solution to access database information.
b. Calculates the accuracy of the system to be built to find out whether the system can translate commands into SQL format and display the requested data.

2. LITERATURE REVIEW

2.1 Previous research

The required tool is a sub-system that can provide answers to questions asked by the user by including questions in Indonesian natural language understood by the user. There are several studies of this toolkit such as research conducted by Setyawan Wibisono in natural language research for academic databases with XML data format by adding sentence patterns that can be the user's reference of the application [1]. Research conducted by Nendi Isharmawan in the study of natural language translator into SQL language with database

limitation used to analyze only one database, and cases that are exemplified to access 1 table [2]. Research conducted by Muhammad Habibi Taqwim in conducting research to build a system that can analyze the story and the answer by applying natural language processing methods, especially the extraction of information in order to produce steps to solve the story to facilitate students to check back the answer to exercise questions there, where in this study the method used can be applied in the case of this translation, because it has the same characteristics that answer the question in Indonesian [3].

Based on the above description, it is necessary to conduct research to build a system that can analyze questions, by applying natural language processing methods, especially the extraction of information in order to generate steps to translate the question into a more dynamic SQL format with multiple table cases.

2.2 Theoretical basis

In this study related to many theories that exist, Natural Language Processing (NLP) is the field of computer science and linguistics related to the interaction between human and computer [4], Rule-based System is a way to store or manipulate knowledge to menintrepentasikan useful information [5], Information Extraction is To obtain structured information from unstructured text, the first thing that needs to be defined is target information as structured information to be extracted [6].

3. TRANSLATION METHOD

The method of translating commands in natural language into the SQL syntax has several stages including, Analyzing the database to get table info, attributes and relationships between tables, then adding the word alias in each table and attribute to enrich the keyword of the table name and its attributes. The next stage of the translation process, before translated sentence will be through the process of Preprocessing ie, tokenizing stage to break the sentence, then stemming stage to remove the word berimbuhan, and alias stage to change the word alias used in the sentence to be the word in question. After that is done the translation process is divided into 5 processes namely, the process of identification of commands, identification tables and relationships, attribute identification, feature identification, and Query preparation process. Last performed the testing process to display the results of execution Syntax SQL form of execution result table. System general process block can be seen in Figure 1.

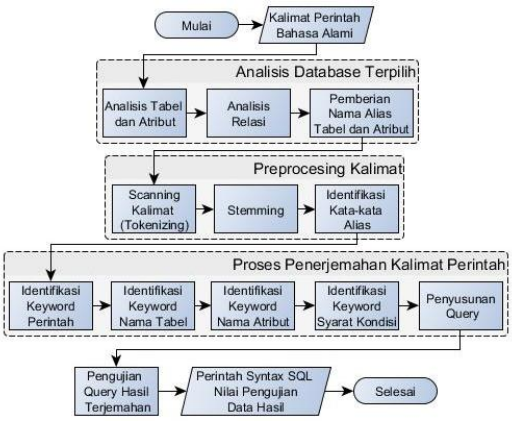


Figure 1. System General Process Block

3.1 Input Command Sentence

In the process of extraction of information, input data in the form of Indonesian command sentences entered by the user. Example of the following command input sentence

" Tampilkan judul film yang termasuk kategori comedy, dimana dalam deskripsi film tersebut mengandung kata "dentist" "

For Examples of Translation Cases Sentences The command selected database is a sakila database [7] which has 23 tables and is related.

3.2 Database Analysis

The earliest stage before entering a command line is to select the database to be accessed and to register the word alias to make it easier to write a command line. Block database analysis process what is seen in Figure 2.

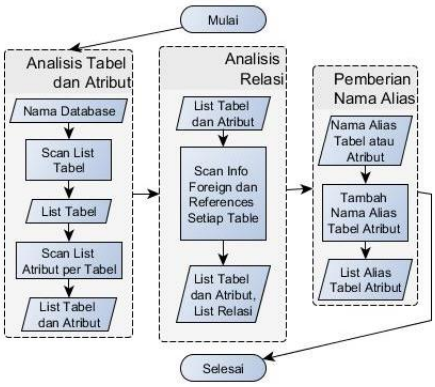


Figure 1. Block Database Analysis Process

In this block the system analyzes the database selected for use. There are 3 processes in this block that is, Analysis Tables and Attributes, Relation Analysis, and Alias Giving. Analysis of Tables and Attributes is the process of identifying all table names and attributes that exist in the database. Relationship Analysis is the process of searching which tables are related. Alias Giving is the process by which users can register the word alias of the table name or attribute that has been previously identified.

a. Analysis of Attribute Table Databases

When the database is selected then the system will scan all the tables contained in the database, then from each scanned table will also be scanned all its attributes, so that the table and its attributes are obtained. The following Table 1 Partial Analysis of Table Name and its attributes.

Table 1. Results of Table and Attribute Name Analysis

#Tbl	#Attr	Name of Table/Atribut	Type
1	**actor**		Base Tabel
	1	actor_id	smallint(5) unsigned
	2	first_name	varchar(45)
	3	last_name	varchar(45)
	4	last_update	timestamp
...	...	...	...
23	4	last_update	timestamp

b. Relation Analysis Table

After the table name and its attributes are scanned, then look for relations between tables that have been scanned by scanning the keyword constraint, foreign key, and references in the schema information using the "show create table <table>" command in each table scanned. Table 2 is an example of analyst relations.

Table 2. Results of Relation Analysis Table

#	Name Relation	*Foreign Key* (Table_name. Attribute_name)	*References* (Table_name. Attribute_name)
1	fk_address_city	address.**city_id**	city.**city_id**
...	...	...	...
22	fk_store_staff	store.**manager_staff_id**	store.**manager_staff_id**

c. Input Alias Name

Input Alias Name in the table or attribute is done by the user and is optional. Table 3 is an example of naming an alias.

Tabel 3. Results of Input Alias Name

#Tbl	#Attr	Tabel Name / Atribut	Alias
1	**actor**		"aktor" "pemain"
	1	actor_id	
	2	first_name	"nama" "aktor"
	3	last_name	"lenkap"
	4	last_update	
...	...	...	...
23	4	last_update	

3.3 Preprocessing Sentences

This preprocessing stage aims to facilitate the application of rule-based methods in solving story problems. In the second block Preprocessing sentences there are 3 processes namely, Tokenizing, Stemming and Identification of the word alias. Tokenizing is breaking sentences into word tokens. Stemming is the basic word search process of every tokenizing token. The Alias word identification process is looking for the alias words listed to be replaced by the table name or attribute in question.

a. *Tokenizing*

In this tokenizing stage a string is cut based on each word of the constituent. Here Figure 3 is a preprocessing tokenizing groove.

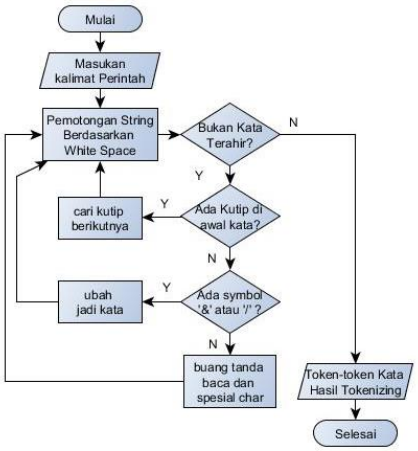


Figure 3. Tokenizing Process Flow

The deduction is done when the input has white space. Special characters other than letters and numbers will be regarded as string composers when the characters stand alone and are separated by spaces. The characters can be seen in the attachment. Table 4 is an example of the tokenizing stage process result in a command line.

Table 4. Result Tokenizing Process

Name	Object	
Input Sentences	Tampilkan judul film yang termasuk kategori comedy, dimana dalam deskripsi film tersebut mengandung kata "dentist"	
Output	Token 1 : Tampilkan Token 2 : judul Token 3 : film Token 4 : yang Token 5 : termasuk Token 6 : kategori Token 7 : comedy Token 8 : dimana	Token 9 : dalam Token 10 : deskripsi Token 11 : film Token 12 : tersebut Token 13 : mengandung Token 14 : kata Token 15 : 'dentist'

In Table 4 above it can be observed that each word and character of the spaced input text input (white space) will be categorized as a token by the system. Every word that uses quotes will be used as each Siantar quotation is a token, and the quotation is quoted as one. For symbols like '&' or '/' will be changed to the word in question. Then for each punctuation will be removed.

b. Stemming

Stemming is the process of converting a word to a base word. Language rules are applied to strip off the affixes. The following is the flow of the application of the stemming process on each tokenizing token.

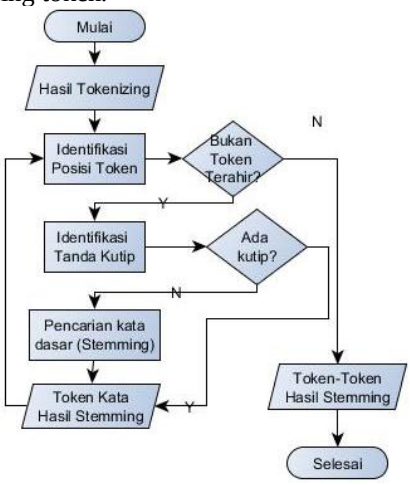


Figure 4. Stemming Process Flow

The stemming process uses literary stemmer which is a simple stemmer library designed to be easy to use. The stemming process is done to every tokenizing result token. The result of this stemming also removes the special characters attached to each token. Table 5 is an example of the stemming stage results of tokens.

Table 5. Results of Stemming Process

Name	Object	
Output	Token 1 : Tampilkan => tampil Token 2 : judul Token 3 : film Token 4 : yang Token 5 : termasuk Token 6 : kategori Token 7 : comedy Token 8 : dimana	Token 9 : dalam Token 10 : deskripsi Token 11 : film Token 12 : tersebut => sebut Token 13 : mengandung => kandung Token 14 : kata Token 15 : 'dentist'

In Table 5 above it can be observed that any attributed token will be replaced by its basic word, unless it is one of the unnecessary distemming lists of words or tokens. Tokens passing through the stemming process, each letter being lowercase.

c. Alias Identification

At this stage do the identification of words alias, to replace these words into the true meaning of the table name or attribute name in question. Here's the process flow.

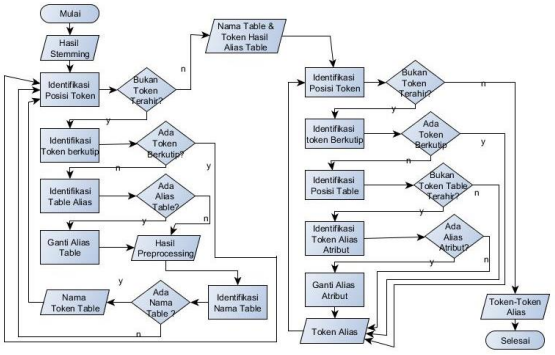


Figure 5. Alias Identification Flow

In this process two repetitions are performed, the first loop to identify the table alias word and replaced with the name of the table in question as well as identify the names of the tables identified in those tokens, then the second loop to look up the alias word attribute of each token based on the list alias attribute of the previously identified table, and replaced also with the attribute name in question. Based on the alias word previously exemplified then the result after identification process of alias word can be seen in Table 6.

Table 6. Results of Alias Identification Process

Name	Object	
Input	Token 1 : tampil Token 2 : judul => title Token 3 : film Token 4 : yang Token 5 : termasuk Token 6 : kategori => category Token 7 : comedy Token 8 : dimana	Token 9 : dalam Token 10 : deskripsi => description Token 11 : film Token 12 : sebut Token 13 : kandung Token 14 : kata Token 15 : 'dentist' => dentist

In Table 6 above we can see there are some alias words, token "title" to "title", token "category" to "category", token "description" to "description", and for quoted toke, quotes are omitted.

3.4 Sentence Translation Process

The result of the preprocessing stage of the sentence is a token list that is ready to be processed at the translation phase of the command line. Here Figure 6 is the Translation Process block of the command line.

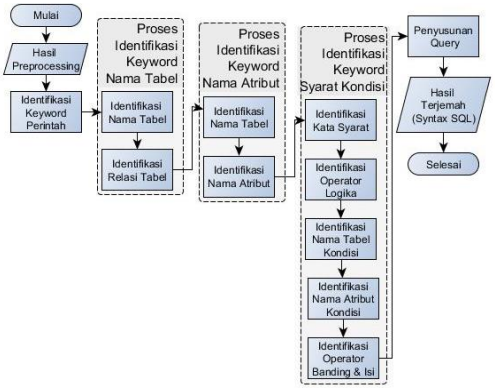


Figure 6. Process Block for Translator Sentence

This stage apply rule based system to get information from the preprocessing process and process it in such a way that can translate the sentence of the command. In this process block there are 5 sub process that is. Identify keyword commands, Identify table names, identify attribute names, identify condition conditions, and construct queries. Identification command is the process of searching for command words in tokent-tokens. Identify table names is the process of finding table names of tokens. Identify the attribute name is the process of finding what attributes to display. Identification of condition conditions is the process of looking for whether there are any signs about using the facility Where conditions. Preparation of queries is a process that collects all the identification results and compiles them in Syntax SQL.

a. Command Identification

words that indicate the use of SELECT are searched in the preprocessing results list in this process. The following Figure 7 is the path of the command keyword identification process.

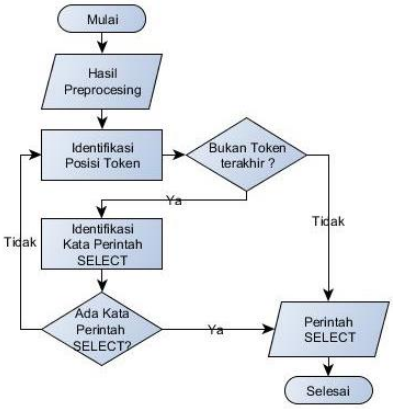


Figure 7. Flow of Command Identification Process

The identification of command words is performed on each preprocessing token. If it is identified then the command variable will be determined according to the group of command words as follows:

Table 7. The word command

No	command	List of Word
1	SELECT	tampil, apa, berapa, siapa, cari, lihat

The following is an example of command-line identification of Preprocessing token tokens.

Table 8. Result of Identification Process of Word Command

Object Name	Object Token	
[Input] Array Token Preprocessing	Token 1 : tampil => SELECT Token 2 : title Token 3 : film Token 4 : yang Token 5 : termasuk Token 6 : category Token 7 : comedy Token 8 : dimana	Token 9 : dalam Token 10 : description Token 11 : film Token 12 : sebut Token 13 : kandung Token 14 : kata Token 15 : dentist
[output] Identification	Ident_perintah : SELECT	

command	

In Table 8 the word "appear" is found in token 1, so the command identified is "SELECT".

b. Identify Table

The command line in this case is always related to the tables in the database, to find out what tables are there in the sentence, then searching the keyword table name of the token results of preprocesing, each table has the possibility of using relations, it is necessary to find relation table.

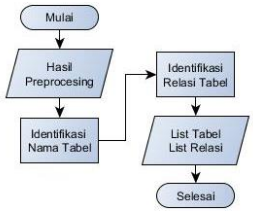


Figure 8. Process Flow of Identifying Tables and Relationships

In this stage two processes are done, namely: the process of identifying table names and the process of identifying table relations. A word token that indicates the table name is searched in the process of identifying table names like Figure 9.

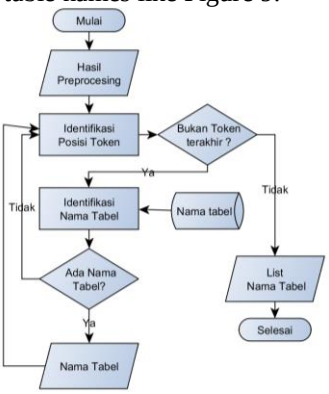


Figure 9. Process Flow Identify Table Name

The process of identifying table names is done to each existing token by comparing the table names that exist in the support database. Each table results table name identification will check the relation as in Figure 10.

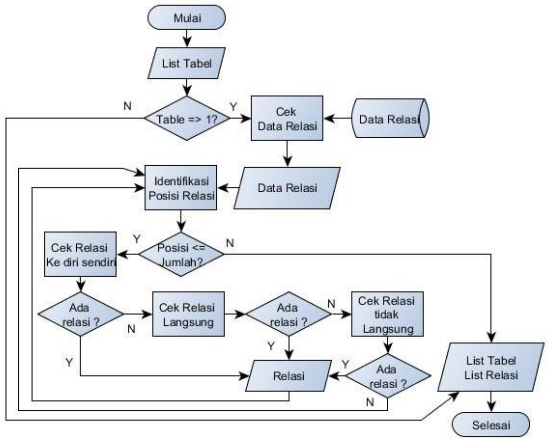


Figure 10. Flow of Relation Identification Process

In this process carried out several checks such as check the relation to yourself, check the relation to the direct, and check the relationship is not direct.

Table 9. Results of Table and Relation Identification Process

Object Name	Object Token	
[Input] Array Token Preprocessing	Token 1 : tampil Token 2 : title Token 3 : film => film Token 4 : yang Token 5 : termasuk Token 6 : category => category Token 7 : comedy Token 8 : dimana	Token 9 : dalam Token 10 : description Token 11 : film => film Token 12 : sebut Token 13 : kandung Token 14 : kata Token 15 : dentist
[output] Identifikasi Tabel	ident_table[1] : film ident_table[2] : category	
[output] Identifikasi Relasi	ident_relasi[1] : FROM `film` ident_relasi[2] : INNER JOIN `film_category` ON `film`.`film_id` = `film_category`.`film_id` ident_relasi[3] : INNER JOIN `category` ON `film_category`.`category_id` = `category`.`category_id`	

c. Identify Attributes

A word token that indicates the attribute name of the identified table is searched in this process.

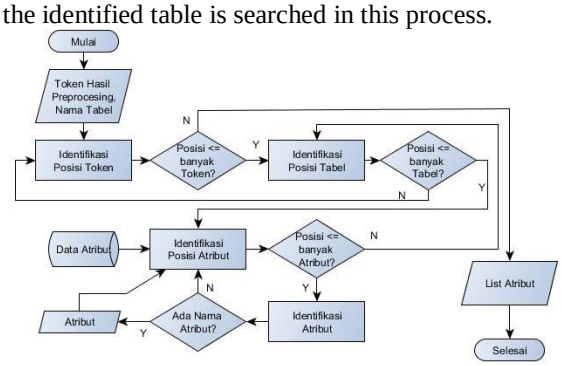


Figure 11. Alur Proses Identifikasi Atribut

In this process the attribute name checking is performed to each token by being compared to each attribute belonging to the identified table.

Table 10. Results of Attribute Identification Process

Object Name	Object Token	
[Input] Identifikasi Tabel	ident_table[1] : film ident_table[2] : category	
[Input] Array Token Preprocessing	Token 1 : tampil Token 2 : title => `title` Token 3 : film Token 4 : yang Token 5 : termasuk Token 6 : category Token 7 : comedy Token 8 : dimana	Token 9 : dalam Token 10 : description => `description` Token 11 : film Token 12 : sebut Token 13 : kandung Token 14 : kata Token 15 : dentist
[output] Identifikasi Atribut	ident_attribute[1] : `film`.`title` AS 'judul' ident_attribute[2] : `film`.`description` AS 'deskripsi'	

In Table 10 found two attributes of the film table.

d. Feature Identification

In this process Stage will look for word tokens that indicate feature where or feature order.

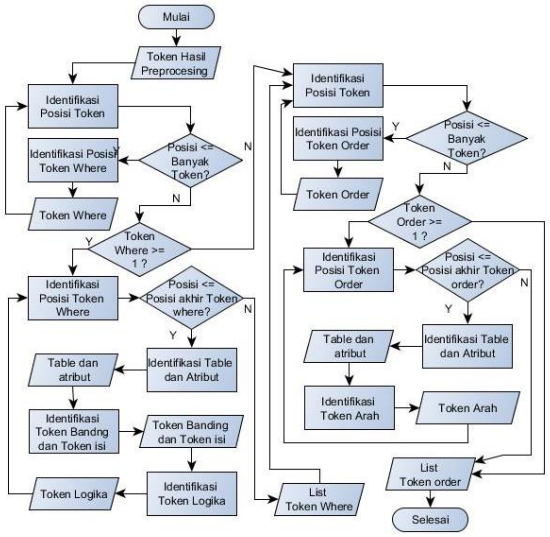


Figure 12. Path of Feature Identification Process

In this process checked the features WHERE and ORDER, each feature to check according to the needs of its features.

Table 11. Feature Process Feature Results

Object Name	Object Token	
[Input] Array Token Preprocessing	Token 1 : tampil Token 2 : title Token 3 : film Token 4 : yang => Fitur Kondisi Token 5 : termasuk Token 6 : category	Token 7 : comedy Token 8 : dimana => Fitur Kondisi Token 9 : dalam Token 10 : description Token 11 : film Token 12 : sebut Token 13 : kandung Token 14 : kata Token 15 : dentist
[output] Identifikasi Kondisi	ident_where[1] : yang ident_where[2] : dimana	
[output] Identifikasi Operator Logika	ident_where_logika[1] : AND	
[output] Identifikasi Tabel Kondisi	ident_where_tabel[1] : category ident_where_tabel[2] : film	
[output] Identifikasi Atribut Kondisi	ident_where_atribut[1] : name ident_where_atribut[2] : description	
[output]	ident_where_banding[1] : =	

Object Name	Object Token
Identifikasi Operator Banding	ident_where_banding[2] : LIKE
[output] Identifikasi Isi Kondisi Operator Logika	ident_where_isi[1] : 'comedy' ident_where_isi[2] : '%dentist%'

In Table 11 we found 2 keywords of WHERE feature, and checked feature requirement, so the requirement to use this feature was fulfilled.

e. Query Arrangement

This stage is the last stage of the translation process, and will check the results of each prior identification process.

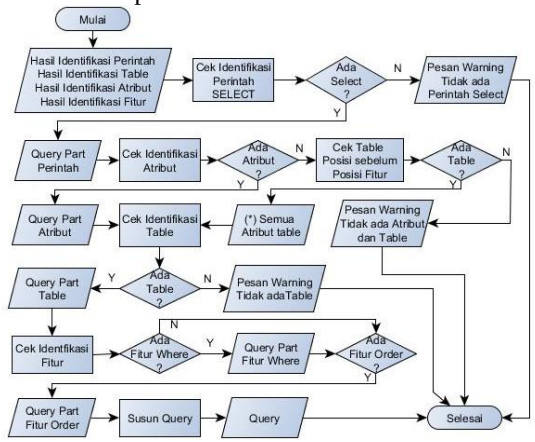


Figure 13. Query Arrangement Process Flow

In this process checking is done on each result of identification, if there are results of identification that does not match it will make the query result is invalid and for some checks will appear error message.

Table 12. Feature Process Feature Results

Object Name	Object Token
[Input] Identifikasi Perintah	Ident_perintah : SELECT
[Input] Identifikasi Tabel	ident_table[1] : film ident_table[2] : category ident_relasi[1] : FROM `film` ident_relasi[2] : INNER JOIN `film_category` ON `film`.`film_id` = `film_category`.`film_id` ident_relasi[3] : INNER JOIN `category` ON `film_category`.`category_id` = `category`.`category_id`
[Input] Identifikasi Atribut	ident_attribute[1] : `title` AS 'judul' ident_attribute[2] : `description` AS 'deskripsi'
[Input] Identifikasi Kondisi	ident_where[1] : yang ident_where[2] : dimana ident_where_logika[1] : AND ident_where_tabel[1] : category ident_where_tabel[2] : film ident_where_atribut[1] : name ident_where_atribut[2] : description ident_where_banding[1] : = ident_where_banding[2] : LIKE ident_where_isi[1] : 'comedy' ident_where_isi[2] : '%dentist% [11]
[output] Penyusunan Query	query_part[1] : SELECT query_part[2] : `film`.`title` AS 'judul' query_part[3] : FROM `film` query_part[4] : INNER JOIN `film_category` ON `film`.`film_id` = `film_category`.`film_id` query_part[5] : INNER JOIN `category` ON `film_category`.`category_id` = `category`.`category_id` query_part[6] : WHERE query_part[7] : `category`.`name` = 'comedy' query_part[8] : AND query_part[9] : `film`.`description` LIKE '%dentist%' query_part[10] : ;
[output] Hasil Query	SELECT `film`.`title` AS 'judul' FROM `film` INNER JOIN `film_category` ON `film`.`film_id` = `film_category`.`film_id` INNER JOIN `category` ON `film_category`.`category_id` = `category`.`category_id` WHERE `category`.`name` = 'comedy' AND `film`.`description` LIKE '%dentist%';

In Table 12 no identification crown is found.

3.4 Query Testing Results

In this last process is to test the results that have been translated into the SQL syntax to be executed syntaxnya and views the data into ouputnya.

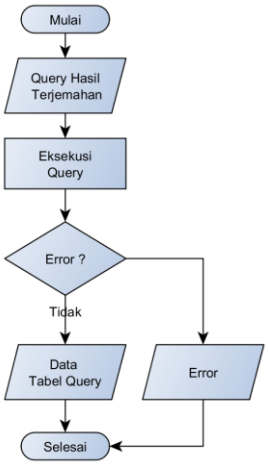


Figure 14. Query Result Testing Flow

If the translation query is invalid it will display an error message.

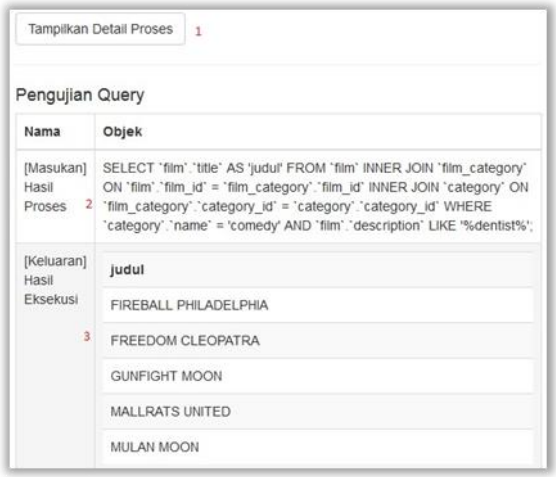


Figure 15. Query Result Test Result Flow

Figure 15 shows the execution result of the translation query.

4. TESTING

This test is done categorization of the types of queries to be tested, here is the query category:

Table 13. Query Type

No	Query Type
1	Query tanpa kondisi
2	Query dengan kondisi
3	Query Banyak kondisi
4	Query banyak tabel tanpa kondisi
5	Query banyak tabel banyak kondisi
6	Query dengan order

The sentences to be tested that have been grouped by category are as follows:

Table 14. Testing Sentence

No	Sentence	Type	The expected query
1	Tampilkan mahasiswa	1	select * from mahasiswa
2	Tampilkan nama mahasiswa	1	select nama from mahasiswa
3	Lihat nilai uas dan uts	1	select uas , uts from nilai
4	sebutkan nama, alamat dan kota mahasiswa	1	select nama, alamat, kota from mahasiswa
5	Lihat mahasiswa yang nimnya 10113005	2	select * from mahasiswa where nim = '10113005'
6	tampilkan nim mahasiswa yang namanya jamal	2	select nim from mahasiswa where nama = 'jamal'
7	tampilkan nama mahasiswa yang jenis kelaminnya perempuan	2	select nama from mahasiswa where jk = 'perempuan'
8	tampilkan mahasiswa berdasarkan jenis kelamin	6	select nama from mahasiswa order by jk
9	tampilkan nilai uas diurutkan berdasarkan paling kecil	6	select * from mahasiswa where uas sort by ascending
10	lihat mahasiswa yang tinggal di bandung dan namanya ahmad	3	select * from mahasiswa where kota = 'bandung' and nama = ahmad
11	tampilkan mahasiswa yang berjenis kelamin perempuan dan tinggal di bandung	3	select * from mahasiswa where jk = 'perempuan' and kota = 'bandung'
12	tampilkan nilai dan kelas	4	select * from nilai join kelas on nilai.kode_kelas = kelas.kode_kelas
13	lihat kelas mahasiswa	4	select * from mahasiswa join kelas on mahasiswa.kode_kelas = kelas.kode_kelas
14	sebutkan nama mahasiswa dan nilai uas	4	select nama , uas from mahasiswa join nilai on mahasiswa.nim = nilai.nim
15	sebutkan nama kelas dan nilai uas	4	select nama_kelas, uas from nilai join kelas on nilai.kode_kelas = kelas.kode_kelas
16	sebutkan mahasiswa yang nilai uasnya 80 sampai 90	5	select * from mahasiswa join nilai on mahasiswa.nim = nilai.nim where uas between 80 and 90
17	siapa mahasiswa yang uas nya bernilai 90	5	select * nama from mahasiswa join nilai on mahasiswa.nim = nilai.nim where uas = '90'
18	Tampilkan judul film yang termasuk kategori comedy, dimana dalam deskripsi film tersebut mengandung kata "dentist"	5	SELECT title AS "Judul Film" FROM film JOIN film_category USING (film_id) JOIN category USING (category_id) WHERE category.name='comedy' AND description LIKE '%dentist%'

The results of the test are as follows:

Table 15. Sentence Testing

No	Test result	Conclusions
1	SELECT * FROM `mahasiswa` ;	Sesuai
2	SELECT `mahasiswa`.`nama` FROM `mahasiswa` ;	Sesuai
3	SELECT `nilai`.`uas` , `nilai`.`uts` FROM `nilai` ;	Sesuai

No	Test result	Conclusions
4	SELECT `mahasiswa`.`nama` , `mahasiswa`.`alamat` , `mahasiswa`.`kota` FROM `mahasiswa` ;	Sesuai
5	SELECT * FROM `mahasiswa` WHERE `mahasiswa`.`nim` = '10113005';	Sesuai
6	SELECT `mahasiswa`.`nim` FROM `mahasiswa` WHERE `mahasiswa`.`nama` = 'jamal';	Sesuai
7	SELECT `mahasiswa`.`nama` FROM `mahasiswa` WHERE `mahasiswa`.`jk` = 'perempuan';	Sesuai
8	tampilkan mahasiswa berdasarkan jenis kelamin	Sesuai
9	SELECT `nilai`.`uas` FROM `nilai` ORDER BY ``.`` DESC;	Kurang sesuai
10	SELECT * FROM `mahasiswa` WHERE `mahasiswa`.`kota` = 'di';	Kurang sesuai
11	SELECT * FROM `mahasiswa` WHERE `mahasiswa`.`kota` = 'di';	Kurang sesuai
12	SELECT * FROM `nilai` INNER JOIN `kelas` ON `nilai`.`kode_kelas` = `kelas`.`kode_kelas` ;	Sesuai
13	SELECT * FROM `kelas` INNER JOIN `nilai` ON `kelas`.`kode_kelas` = `nilai`.`kode_kelas` INNER JOIN `mahasiswa` ON `nilai`.`nim` = `mahasiswa`.`nim` ;	Sesuai
14	SELECT `mahasiswa`.`nama` , `nilai`.`uas` FROM `mahasiswa` INNER JOIN `nilai` ON `mahasiswa`.`nim` = `nilai`.`nim` ;	Sesuai
15	SELECT `kelas`.`nama_kelas` AS 'nama' , `nilai`.`uas` FROM `kelas` INNER JOIN `nilai` ON `kelas`.`kode_kelas` = `nilai`.`kode_kelas` ;	Sesuai
16	SELECT * FROM `mahasiswa` INNER JOIN `nilai` ON `mahasiswa`.`nim` = `nilai`.`nim` WHERE `nilai`.`uas` = '80';	Kurang sesuai
17	SELECT * FROM `mahasiswa` WHERE ;	Tidak sesuai
18	SELECT `film`.`title` AS 'judul' FROM `film` INNER JOIN `film_category` ON `film`.`film_id` = `film_category`.`film_id` INNER JOIN `category` ON `film_category`.`category_id` = `category`.`category_id` WHERE `category`.`name` = 'comedy' AND `film`.`description` LIKE '%dentist%';	Sesuai

The accuracy test used in this research applied F - Measure method. F-Measure is a representation of the measure of the accuracy of a test by using the precision and recall values of the test [7].

This test looks at whether the system can process sentences entered by the user to be translated so that it can be measured accuracy with the method of F - Measure. the sentence entered has been translated manually first so it can be compared with the output of the system. The values calculated in this test are the correct, falsePositive, falseNegative, precision, recall, and F - Measure values of the system dataset iteration. Calculation F - Measure microaverage data by using total correct value, falsePositive, falseNegative overall dataset while calculation while F - Measure macroaverage is by calculating average F - Measure each iteration dataset. Here is the result of iterating the entire dataset of system accuracy testing.

$$Dataset\ Kalimat$$
$$Total\ Dataset = TD = 18$$
$$Total\ nilai\ correct = Tnc = 13$$
$$total\ nilai\ falsePositive = tnfP = 4$$
$$total\ nilai\ falseNegative = tnfN = 1$$

$$Precision\ micro = \frac{Tnc}{(Tnc + tnfP)} = \frac{13}{(13+4)} = \frac{13}{17} = 0.7647 \quad (1)$$

$$Recall\ micro = \frac{Tnc}{(Tnc + tnfN)} = \frac{13}{(13+1)} = \frac{13}{14} = 0.9286 \quad (2)$$

$$F - Measure\ microaverage = \frac{2\ x\ \left(\frac{Precision}{micro}\right)\ x\ \left(\frac{Recall}{micro}\right)}{\left(\frac{Precision}{micro}\right) + \left(\frac{Recall}{micro}\right)} = \frac{2\ x\ (0.7647)\ x\ (0.9286)}{(0.7647) + (0.9286)} = \frac{1.4202}{1.6933} \quad (3)$$

$$= 0.8387\ X\ 100\% = 83.87\%$$

$$F - Measure\ microaverage = \sum_{i=0}^{n=40} F - Measure\ microaverage\ x100\% = \frac{1+1+0+0+\cdots+0+1+1}{40} x\ 100\% \quad (4)$$

$$= \frac{13}{18} x\ 100\% = 0.7222\ x\ 100\% = 72.22\%$$

So the system accuracy based on the test dataset above is: The accuracy value is known to be between 72.22% - 83.87%.

5. CLOSING

3.1 Conclusions

After analyzing, designing, and testing using blackbox method and accuracy testing, it was obtained that the system can translate the command line in natural language into SQL query syntax format. Test results from several sentences obtained accuracy ranged from 72.22% - 83.87%. indicates that when the sentences entered by the user is incomplete it will also get the wrong query or produce the wrong information. This can result in an error in executing the query. Thus the information seeker media can directly access the source information on the database when the data in the database is changed, then the results of the displayed data will also change.

3.2 Suggestions

For better research development, there are some suggestions for improving system capabilities and adding other features as follows:

- Make the system a non-application library with Analyze tools and alias form used as a configuration for use in other applications.
- Added features to resolve related words like place name, full name, a sentence, date and so on.
- Added features of Aggregation function, DATE TIME, DISTINCT, GROUP BY, HAVING, LIMIT, SUB QUERY, SELECT expressions.
- Adding rules for relations other than INNER JOIN, rules for more dynamic LIKE features, such as various letters etc., rules for possible validation of errors, sentence translation rules to SQL queries in order to improve the accuracy of system key answers.
- The addition of more varied Test questions to test all possibilities.
- Development in wider coverage such as DML, DDL, DCL, and so on.

BIBLIOGRAPHY

[1] S. Wibisono, Aplikasi Pengolahan Bahasa Alami untuk Query Basisdata Akademik dengan Format Data XML, Skripsi S1. Teknik Informatika. Universitas Stikubank, 2013.

[2] N. Isharmawan, Ekstraksi Informasi dan SQL Parser untuk Query SQL dari Teks Berbahasa Indonesia, Skripsi S1. Teknik Informatika. Universitas Komputer Indonesia, 2017.

[3] M. H. Taqwim, Ekstraksi Informasi dengan Metode Rule – Based untuk Evaluasi Pemahaman Fisika Kinematika, Skripsi S1. Teknik Informatika. Universitas Komputer Indonesia, 2016.

[4] S. Entin, "Kecerdasan Buatan," 28 September 2011. [Online]. Available: http://entin.lecturer.pens.ac.id/Kecerdasan%20Buatan/Buku/. [Accessed 19 April 2017].

[5] A. Ismaya, Algoritma Ekstraksi Informasi Berbasis Aturan, Yogyakarta: Skripsi S1. Teknik Informatika : Universitas Gajah Mada, 2014.

[6] M. L. Khodra, Ekstraksi Informasi Transaksi Online pada Twitter, Bandung: Skripsi S1. Teknik Informatika : Institut Teknologi Bandung, 2013.

[7] MySQL Documentation "Sakila Sample Database,". [Online]. Available: https://dev.mysql.com/doc/sakila/en/ [Accessed 24 Februari 2018].

[8] H. Ruwdanar, R. Arum SP. Dan H. Fitriawan, Rancang Bangun Sistem Informasi Akademik Menggunakan Php Dan Mysql, Bandar Lampung: Universitas Negeri Lampung, 2012.