소프트웨어무결점연구센터 제11차 워크샵, 2014년 여름

데이터베이스 스키마 재구성을 이용한 SQL 문 정적검사

조소희[1], 도경구[1]
[1]한양대학교 프로그래밍언어연구실
{jsh, doh}@hanyang.ac.kr

학술

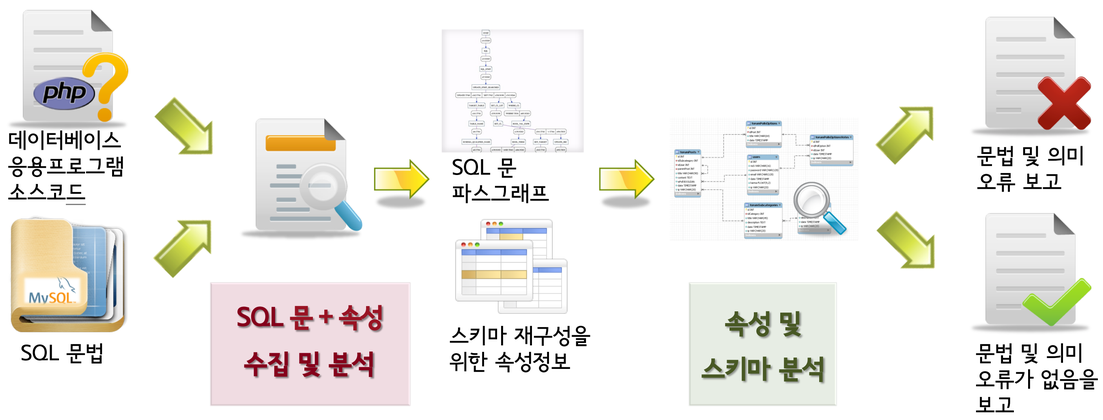
1. 목표: 문법검사+의미검사
2. 대상: PHP+MySQL 기반 응용프로그램
3. 문법 및 의미검사 대상 항목(MySQL 기준[1],[2])
   ① Cardinality Violation
   ② Data Exception
   ③ Syntax Error or Access Rule Violation
4. 방법(구조도 참조)
   - 요약파싱과 속성문법을 이용한 SQL 문 분석
   - 데이터베이스 스키마 재구성을 이용한 의미분석
5. 결론
   ① 요약파서가 파스그래프를 생성하는 과정에서 속성을 이용하여 문법 및 의미검사 가능
   ② 기존연구들은 데이터베이스 스키마가 이미 주어진 상태에서 타입을 알아내거나[3],[4] DBMS에 실제로 질의를 보낸 결과로 의미오류를 판단[5]
   ➔ 응용프로그램 소스코드만을 가지고 제한적으로나마 의미오류를 판단
6. 향후 연구
   ① 데이터베이스 스키마와 응용프로그램 간의 양방향 영향분석
      - 응용프로그램 소스코드 정보를 담고 있는 재구성한 데이터베이스 스키마와 실제 DBMS 간의 매핑을 이용
      - SQL 문에 영향을 주는 부분과 질의 결과가 영향을 주는 범위 확대 분석
      - 변경유형에 따른 가중치를 부과하여 변경 시 영향정도 분석
   ② 응용프로그램에서 사용하는 데이터베이스 스키마 접근권한 최소화를 위한 분석
      - 재구성한 스키마 정보를 데이터베이스 연결정보에 따라 나눔
      - 데이터베이스 사용자 및 응용프로그램 사용자 별 접근범위를 분석하여 최소단위 접근권한을 부여
7. 참조

[1] ISO/IEC 9075-2:2008(24.1 SQLSTATE)
[2] MySQL 5.5 Reference Manual(Appendix B.3 Server Error Codes and Messages)
[3] G. Wassermann, et. al, Static checking of dynamically generated queries in database applications, *ACM Trans. Softw. Eng. Methodol.*, pp 14:1-27, 2007
[4] A. Dasgupta, et. al, A Static Analysis Framework for Database Applications, *Proceedings. of IEEE the 25th International Conference on Data Engineering*, pp. 1403-1414, 2009
[5] A. Annamaa, et. al, An Interactive Tool for Analyzing Embedded SQL Queries, *Programming Languages and Systems*, pp 131-138, 2010

요약파싱과 속성문법을 이용한 SQL 문 분석

입력: 데이터베이스 응용프로그램과 SQL 문법
출력: 핫스팟마다의 SQL 문 파스그래프,
정적검사와 스키마 재구성을 위한 속성정보

1. PHP 소스를 파싱하여 흐름방정식으로 변환
2. 핫스팟(DBMS로 SQL 문을 보내는 부분,
   예> `query($conn, $sql);`에서 `$sql` 부분)에 영향을 주는 흐름방정식만 남기고 잘라냄
3. 변환 과정에서 SQL 문과 연관이 있는 함수 호출 부분도 함께 분석해서 스키마 관련 정보를 얻어냄
   예> bind_param("ds", $age, $email);
4. 흐름방정식을 SQL 문법에 맞는지 요약파싱을 통해 검증
   - 요약파싱 과정에서 데이터베이스 스키마 관련 정보를 함께 모으고 오류가 발생하지 않으면 SQL 문 파스그래프와 함께 다음 단계로 넘겨줌
   - 요약파싱 결과 에러가 발생하면 문법오류로 보고
   - 속성분석 시 오류가 발생하면 의미오류로 보고



데이터베이스 응용프로그램 SQL 문 정적검사 흐름도

데이터베이스 스키마 재구성을 이용한 의미분석

입력: SQL 문 파스그래프,
정적검사와 스키마 재구성을 위한 속성정보
출력: SQL 문 문법 및 의미 검사 결과

1. 여러 핫스팟에서 모은 속성정보를 분석해서 데이터베이스 스키마 정보를 재구성
2. 재구성 과정에서 같은 스키마 객체에 대해 컬럼 타입, 제약조건 등이 상충되는 경우 속성이 수집된 위치의 파스그래프 분석 후 해당 의미오류 보고

의미-TYPE

MySQL Error #1222 (ER_WRONG_NUMBER_OF_COLUMNS_IN_SELECT)

```
SELECT id, type, details
FROM Contacts
UNION
SELECT 3, 'Consulting'
```

>> Error Code: 1222. The used SELECT statements have a different number of columns

1. SELECT → stmt=SELECT, cl=SELECT
2. id → id="id"
3. column_ref → col = {"id", "", UNKNOWN}
4. select_lst →
   select_cols = [{"id", "", UNKNOWN}; {"type", "", UNKNOWN};
   {"details", "", UNKNOWN};]
5. FROM → cl=FROM
6. table_name → tab = {"Contacts", "", []}
7. tbl_ref_lst → tabs = [{"Contacts", "", []};]
8. query_spec → (SELECT, 1) -> {select_cols, tabs, …}
9. …중략… // 두 번째 SELECT 문에서도 동일한 과정 반복
10. query_spec→ (SELECT, 2) -> {select_cols, tabs, …}
11. query_expr_body(UNION) →
    s1 = (SELECT, 1)
    s2 = (SELECT, 2)
    cardinality_check(s1.select_cols, s2.select_cols)
    ➔ **ERROR** (|s1.select_cols| != |s2.select_cols|)

의미-NAME

MySQL Error #1056 (ER_WRONG_GROUP_FIELD)

```
SELECT word_name,
       SUM(word_name) AS word_count
FROM Words
GROUP BY word_count
```

>> Error Code: 1056. Can't group on 'word_count'

1. SELECT → stmt=SELECT, cl=SELECT
2. id → id="word_name"
3. column_ref → col = {"word_name", "", UNKNOWN}
4. aggregate_func → col = {"SUM(word_name)", "", AGGFUN}
5. derived_col →
   col = {"SUM(word_name)", "word_count", AGGFUN}
6. select_lst → select_cols = [{"word_name", "", UNKNOWN};
   {"SUM(word_name)", "word_count", AGGFUN};]
7. …중략… // #1222와 동일한 FROM 절 과정
8. GROUP BY → cl=GROUP_BY
9. id → id="word_count"
10. grpg_elem → col = {"word_count", "", UNKNOWN}
11. grpg_elem_lst → grp_cols = [{"word_count", "", UNKNOWN};]
12. query_spec → (SELECT, 1) -> {select_cols, tabs, grp_cols, …}
    group_field_check(select_cols, grp_cols)
    ➔ **ERROR** (grp_cols.(i) ∈ filter(is_aggfun, select_cols))

의미-UNIQUENESS

MySQL Error #1066 ER_NONUNIQ_TABLE

```
SELECT Film.title AS title, Lang.name AS original,
    Lang.name AS other
FROM Film
JOIN Lang ON Film.org_lang_id = Lang.lang_id
JOIN Lang ON Film.lang_id = Lang.lang_id
```

>> Error Code: 1066. Not unique table/alias: 'Lang'

1. SELECT → stmt=SELECT, cl=SELECT
2. id → id="title"
3. schema_qualified_name = {"title"; ["Film"]}
4. column_ref → col = {"title", "", Table ["Film"]}
5. derived_col → col = {"title", "title", Table ["Film"]}
6. …중략… // 각 컬럼에 대해 동일한 과정 반복
7. select_lst →
   select_cols = [{"title", "title", Table ["Film"]};
   {"name", "original", Table ["Lang"]};
   {"name", "other", Table ["Lang"]}]
8. FROM → cl=FROM
9. table_name → tab = {"Film", "", []}
10. table_name → tab = {"Lang", "", []}
11. table_name → tab = {"Lang", "", []}
12. tbl_ref_lst →
    tabs = [{"Film", "", []}; {"Lang", "", []}; {"Lang", "", []};]
13. query_spec → (SELECT, 1) -> {select_cols, tabs, …}
    check_qualified_table_name(select_cols, tabs)
    ➔ **ERROR** (|filter(match_name(select_cols.(2), tabs.(i)),tabs)|>1,
    |filter(match_name(select_cols.(3), tabs.(i)),tabs)|>1)

◆ 스키마 재구성을 위한 속성 예

✓ joined_table → join_tables = [JOIN(Tabs.(1), Tabs.(2));
JOIN(join_tables.(1), Tabs.(3))]

✓ join_spec (ON)→
join_specs = [{join_tables.(1), ON(EQ,on_cols.(1),on_cols.(2))};…]
➔ type(on_cols.(1))==type(on_cols.(2))

Film

Column Name	Type
film_id	int
title	VARCHAR(200)
original	CHAR(10)
other	Int

Lang

Column Name	Type
lang_id	int
name	VARCHAR(50)

스키마 재구성을 이용한 의미분석 예

Create 문을 분석해서 재구성한 스키마가 왼쪽 그림과 같다면
<의미-UNIQUENESS> 예제 마지막 부분에서 기술한 속성과 상충

➔ ERROR (type(original) <> type(lang_id))

✓ MySQL은 자동형변환이 일어나서 이런 경우 오류로 보고 하지 않은 채 변환된 값이 같은 행에 대해서만 Join한 결과를 줌

✓ 개발자의 의도와 다를 수 있으므로 오류로 보고해야 함