Atitit sql解析关键词 列表

## public class Keywords {

package com.alibaba.druid.sql.parser;

public class Keywords {

public Map<String, Token> getKeywords() {  
 return this.keywords;  
}  
  
static {  
 Map<String, Token> map = new HashMap();  
 map.put("ALL", Token.ALL);  
 map.put("ALTER", Token.ALTER);  
 map.put("AND", Token.AND);  
 map.put("ANY", Token.ANY);  
 map.put("AS", Token.AS);  
 map.put("ENABLE", Token.ENABLE);  
 map.put("DISABLE", Token.DISABLE);  
 map.put("ASC", Token.ASC);  
 map.put("BETWEEN", Token.BETWEEN);  
 map.put("BY", Token.BY);  
 map.put("CASE", Token.CASE);  
 map.put("CAST", Token.CAST);  
 map.put("CHECK", Token.CHECK);  
 map.put("CONSTRAINT", Token.CONSTRAINT);  
 map.put("CREATE", Token.CREATE);  
 map.put("DATABASE", Token.DATABASE);  
 map.put("DEFAULT", Token.DEFAULT);  
 map.put("COLUMN", Token.COLUMN);  
 map.put("TABLESPACE", Token.TABLESPACE);  
 map.put("PROCEDURE", Token.PROCEDURE);  
 map.put("FUNCTION", Token.FUNCTION);  
 map.put("DELETE", Token.DELETE);  
 map.put("DESC", Token.DESC);  
 map.put("DISTINCT", Token.DISTINCT);  
 map.put("DROP", Token.DROP);  
 map.put("ELSE", Token.ELSE);  
 map.put("EXPLAIN", Token.EXPLAIN);  
 map.put("EXCEPT", Token.EXCEPT);  
 map.put("END", Token.END);  
 map.put("ESCAPE", Token.ESCAPE);  
 map.put("EXISTS", Token.EXISTS);  
 map.put("FOR", Token.FOR);  
 map.put("FOREIGN", Token.FOREIGN);  
 map.put("FROM", Token.FROM);  
 map.put("FULL", Token.FULL);  
 map.put("GROUP", Token.GROUP);  
 map.put("HAVING", Token.HAVING);  
 map.put("IN", Token.IN);  
 map.put("INDEX", Token.INDEX);  
 map.put("INNER", Token.INNER);  
 map.put("INSERT", Token.INSERT);  
 map.put("INTERSECT", Token.INTERSECT);  
 map.put("INTERVAL", Token.INTERVAL);  
 map.put("INTO", Token.INTO);  
 map.put("IS", Token.IS);  
 map.put("JOIN", Token.JOIN);  
 map.put("KEY", Token.KEY);  
 map.put("LEFT", Token.LEFT);  
 map.put("LIKE", Token.LIKE);  
 map.put("LOCK", Token.LOCK);  
 map.put("MINUS", Token.MINUS);  
 map.put("NOT", Token.NOT);  
 map.put("NULL", Token.NULL);  
 map.put("ON", Token.ON);  
 map.put("OR", Token.OR);  
 map.put("ORDER", Token.ORDER);  
 map.put("OUTER", Token.OUTER);  
 map.put("PRIMARY", Token.PRIMARY);  
 map.put("REFERENCES", Token.REFERENCES);  
 map.put("RIGHT", Token.RIGHT);  
 map.put("SCHEMA", Token.SCHEMA);  
 map.put("SELECT", Token.SELECT);  
 map.put("SET", Token.SET);  
 map.put("SOME", Token.SOME);  
 map.put("TABLE", Token.TABLE);  
 map.put("THEN", Token.THEN);  
 map.put("TRUNCATE", Token.TRUNCATE);  
 map.put("UNION", Token.UNION);  
 map.put("UNIQUE", Token.UNIQUE);  
 map.put("UPDATE", Token.UPDATE);  
 map.put("VALUES", Token.VALUES);  
 map.put("VIEW", Token.VIEW);  
 map.put("SEQUENCE", Token.SEQUENCE);  
 map.put("TRIGGER", Token.TRIGGER);  
 map.put("USER", Token.USER);  
 map.put("WHEN", Token.WHEN);  
 map.put("WHERE", Token.WHERE);  
 map.put("XOR", Token.XOR);  
 map.put("OVER", Token.OVER);  
 map.put("TO", Token.TO);  
 map.put("USE", Token.USE);  
 map.put("REPLACE", Token.REPLACE);  
 map.put("COMMENT", Token.COMMENT);  
 map.put("COMPUTE", Token.COMPUTE);  
 map.put("WITH", Token.WITH);  
 map.put("GRANT", Token.GRANT);  
 map.put("REVOKE", Token.REVOKE);  
 map.put("WHILE", Token.WHILE);  
 map.put("DO", Token.DO);  
 map.put("DECLARE", Token.DECLARE);  
 map.put("LOOP", Token.LOOP);  
 map.put("LEAVE", Token.LEAVE);  
 map.put("ITERATE", Token.ITERATE);  
 map.put("REPEAT", Token.REPEAT);  
 map.put("UNTIL", Token.UNTIL);  
 map.put("OPEN", Token.OPEN);  
 map.put("CLOSE", Token.CLOSE);  
 map.put("CURSOR", Token.CURSOR);  
 map.put("FETCH", Token.FETCH);  
 map.put("OUT", Token.OUT);  
 map.put("INOUT", Token.INOUT);  
 map.put("LIMIT", Token.LIMIT);  
 DEFAULT\_KEYWORDS = new Keywords(map);  
 Map<String, Token> sqlitemap = new HashMap();  
 sqlitemap.putAll(DEFAULT\_KEYWORDS.getKeywords());  
 sqlitemap.put("LIMIT", Token.LIMIT);  
 SQLITE\_KEYWORDS = new Keywords(sqlitemap);

## public enum Token {

//  
// Source code recreated from a .class file by IntelliJ IDEA  
// (powered by Fernflower decompiler)  
//  
  
package com.alibaba.druid.sql.parser;  
  
public enum Token {  
 SELECT("SELECT"),  
 DELETE("DELETE"),  
 INSERT("INSERT"),  
 UPDATE("UPDATE"),  
 FROM("FROM"),  
 HAVING("HAVING"),  
 WHERE("WHERE"),  
 ORDER("ORDER"),  
 BY("BY"),  
 GROUP("GROUP"),  
 INTO("INTO"),  
 AS("AS"),  
 CREATE("CREATE"),  
 ALTER("ALTER"),  
 DROP("DROP"),  
 SET("SET"),  
 NULL("NULL"),  
 NOT("NOT"),  
 DISTINCT("DISTINCT"),  
 TABLE("TABLE"),  
 TABLESPACE("TABLESPACE"),  
 VIEW("VIEW"),  
 SEQUENCE("SEQUENCE"),  
 TRIGGER("TRIGGER"),  
 USER("USER"),  
 INDEX("INDEX"),  
 SESSION("SESSION"),  
 PROCEDURE("PROCEDURE"),  
 FUNCTION("FUNCTION"),  
 PRIMARY("PRIMARY"),  
 KEY("KEY"),  
 DEFAULT("DEFAULT"),  
 CONSTRAINT("CONSTRAINT"),  
 CHECK("CHECK"),  
 UNIQUE("UNIQUE"),  
 FOREIGN("FOREIGN"),  
 REFERENCES("REFERENCES"),  
 EXPLAIN("EXPLAIN"),  
 FOR("FOR"),  
 IF("IF"),  
 SORT("SORT"),  
 ALL("ALL"),  
 UNION("UNION"),  
 EXCEPT("EXCEPT"),  
 INTERSECT("INTERSECT"),  
 MINUS("MINUS"),  
 INNER("INNER"),  
 LEFT("LEFT"),  
 RIGHT("RIGHT"),  
 FULL("FULL"),  
 OUTER("OUTER"),  
 JOIN("JOIN"),  
 ON("ON"),  
 SCHEMA("SCHEMA"),  
 CAST("CAST"),  
 COLUMN("COLUMN"),  
 USE("USE"),  
 DATABASE("DATABASE"),  
 TO("TO"),  
 AND("AND"),  
 OR("OR"),  
 XOR("XOR"),  
 CASE("CASE"),  
 WHEN("WHEN"),  
 THEN("THEN"),  
 ELSE("ELSE"),  
 ELSIF("ELSIF"),  
 END("END"),  
 EXISTS("EXISTS"),  
 IN("IN"),  
 CONTAINS("CONTAINS"),  
 RLIKE("RLIKE"),  
 FULLTEXT("FULLTEXT"),  
 NEW("NEW"),  
 ASC("ASC"),  
 DESC("DESC"),  
 IS("IS"),  
 LIKE("LIKE"),  
 ESCAPE("ESCAPE"),  
 BETWEEN("BETWEEN"),  
 VALUES("VALUES"),  
 INTERVAL("INTERVAL"),  
 LOCK("LOCK"),  
 SOME("SOME"),  
 ANY("ANY"),  
 TRUNCATE("TRUNCATE"),  
 RETURN("RETURN"),  
 TRUE("TRUE"),  
 FALSE("FALSE"),  
 LIMIT("LIMIT"),  
 KILL("KILL"),  
 IDENTIFIED("IDENTIFIED"),  
 PASSWORD("PASSWORD"),  
 ALGORITHM("ALGORITHM"),  
 DUAL("DUAL"),  
 BINARY("BINARY"),  
 SHOW("SHOW"),  
 REPLACE("REPLACE"),  
 BITS,  
 WHILE("WHILE"),  
 DO("DO"),  
 LEAVE("LEAVE"),  
 ITERATE("ITERATE"),  
 REPEAT("REPEAT"),  
 UNTIL("UNTIL"),  
 OPEN("OPEN"),  
 CLOSE("CLOSE"),  
 OUT("OUT"),  
 INOUT("INOUT"),  
 EXIT("EXIT"),  
 UNDO("UNDO"),  
 SQLSTATE("SQLSTATE"),  
 CONDITION("CONDITION"),  
 DIV("DIV"),  
 WINDOW("WINDOW"),  
 OFFSET("OFFSET"),  
 ROW("ROW"),  
 ROWS("ROWS"),  
 ONLY("ONLY"),  
 FIRST("FIRST"),  
 NEXT("NEXT"),  
 FETCH("FETCH"),  
 OF("OF"),  
 SHARE("SHARE"),  
 NOWAIT("NOWAIT"),  
 RECURSIVE("RECURSIVE"),  
 TEMPORARY("TEMPORARY"),  
 TEMP("TEMP"),  
 UNLOGGED("UNLOGGED"),  
 RESTART("RESTART"),  
 IDENTITY("IDENTITY"),  
 CONTINUE("CONTINUE"),  
 CASCADE("CASCADE"),  
 RESTRICT("RESTRICT"),  
 USING("USING"),  
 CURRENT("CURRENT"),  
 RETURNING("RETURNING"),  
 COMMENT("COMMENT"),  
 OVER("OVER"),  
 TYPE("TYPE"),  
 ILIKE("ILIKE"),  
 START("START"),  
 PRIOR("PRIOR"),  
 CONNECT("CONNECT"),  
 WITH("WITH"),  
 EXTRACT("EXTRACT"),  
 CURSOR("CURSOR"),  
 MODEL("MODEL"),  
 MERGE("MERGE"),  
 MATCHED("MATCHED"),  
 ERRORS("ERRORS"),  
 REJECT("REJECT"),  
 UNLIMITED("UNLIMITED"),  
 BEGIN("BEGIN"),  
 EXCLUSIVE("EXCLUSIVE"),  
 MODE("MODE"),  
 WAIT("WAIT"),  
 ADVISE("ADVISE"),  
 SYSDATE("SYSDATE"),  
 DECLARE("DECLARE"),  
 EXCEPTION("EXCEPTION"),  
 GRANT("GRANT"),  
 REVOKE("REVOKE"),  
 LOOP("LOOP"),  
 GOTO("GOTO"),  
 COMMIT("COMMIT"),  
 SAVEPOINT("SAVEPOINT"),  
 CROSS("CROSS"),  
 PCTFREE("PCTFREE"),  
 INITRANS("INITRANS"),  
 MAXTRANS("MAXTRANS"),  
 INITIALLY("INITIALLY"),  
 ENABLE("ENABLE"),  
 DISABLE("DISABLE"),  
 SEGMENT("SEGMENT"),  
 CREATION("CREATION"),  
 IMMEDIATE("IMMEDIATE"),  
 DEFERRED("DEFERRED"),  
 STORAGE("STORAGE"),  
 MINEXTENTS("MINEXTENTS"),  
 MAXEXTENTS("MAXEXTENTS"),  
 MAXSIZE("MAXSIZE"),  
 PCTINCREASE("PCTINCREASE"),  
 FLASH\_CACHE("FLASH\_CACHE"),  
 CELL\_FLASH\_CACHE("CELL\_FLASH\_CACHE"),  
 NONE("NONE"),  
 LOB("LOB"),  
 STORE("STORE"),  
 CHUNK("CHUNK"),  
 CACHE("CACHE"),  
 NOCACHE("NOCACHE"),  
 LOGGING("LOGGING"),  
 NOCOMPRESS("NOCOMPRESS"),  
 KEEP\_DUPLICATES("KEEP\_DUPLICATES"),  
 EXCEPTIONS("EXCEPTIONS"),  
 PURGE("PURGE"),  
 COMPUTE("COMPUTE"),  
 ANALYZE("ANALYZE"),  
 OPTIMIZE("OPTIMIZE"),  
 TOP("TOP"),  
 ARRAY("ARRAY"),  
 DISTRIBUTE("DISTRIBUTE"),  
 EOF,  
 ERROR,  
 IDENTIFIER,  
 HINT,  
 VARIANT,  
 LITERAL\_INT,  
 LITERAL\_FLOAT,  
 LITERAL\_HEX,  
 LITERAL\_CHARS,  
 LITERAL\_NCHARS,  
 LITERAL\_ALIAS,  
 LINE\_COMMENT,  
 MULTI\_LINE\_COMMENT,  
 BINARY\_FLOAT,  
 BINARY\_DOUBLE,  
 PARTITION,  
 PARTITIONED,  
 OVERWRITE,  
 SEL("SEL"),  
 LOCKING("LOCKING"),  
 ACCESS("ACCESS"),  
 VOLATILE("VOLATILE"),  
 MULTISET("MULTISET"),  
 POSITION("POSITION"),  
 RANGE\_N("RANGE\_N"),  
 FORMAT("FORMAT"),  
 QUALIFY("QUALIFY"),  
 MOD("MOD"),  
 CONCAT("CONCAT"),  
 UPSERT("UPSERT"),  
 LPAREN("("),  
 RPAREN(")"),  
 LBRACE("{"),  
 RBRACE("}"),  
 LBRACKET("["),  
 RBRACKET("]"),  
 SEMI(";"),  
 COMMA(","),  
 DOT("."),  
 DOTDOT(".."),  
 DOTDOTDOT("..,"),  
 EQ("="),  
 GT(">"),  
 LT("<"),  
 LT\_SUB\_GT("<->"),  
 BANG("!"),  
 BANGBANG("!!"),  
 BANG\_TILDE("!~"),  
 BANG\_TILDE\_STAR("!~\*"),  
 TILDE("~"),  
 TILDE\_STAR("~\*"),  
 TILDE\_EQ("~="),  
 QUES("?"),  
 QUESQUES("??"),  
 QUESBAR("?|"),  
 QUESAMP("?&"),  
 COLON(":"),  
 COLONCOLON("::"),  
 COLONEQ(":="),  
 EQEQ("=="),  
 EQGT("=>"),  
 LTEQ("<="),  
 LTEQGT("<=>"),  
 LTGT("<>"),  
 GTEQ(">="),  
 BANGEQ("!="),  
 BANGGT("!>"),  
 BANGLT("!<"),  
 AMPAMP("&&"),  
 BARBAR("||"),  
 BARBARSLASH("||/"),  
 BARSLASH("|/"),  
 PLUS("+"),  
 SUB("-"),  
 SUBGT("->"),  
 SUBGTGT("->>"),  
 STAR("\*"),  
 SLASH("/"),  
 AMP("&"),  
 BAR("|"),  
 CARET("^"),  
 CARETEQ("^="),  
 PERCENT("%"),  
 LTLT("<<"),  
 GTGT(">>"),  
 MONKEYS\_AT("@"),  
 MONKEYS\_AT\_AT("@@"),  
 POUND("#"),  
 POUNDGT("#>"),  
 POUNDGTGT("#>>"),  
 MONKEYS\_AT\_GT("@>"),  
 LT\_MONKEYS\_AT("<@");  
  
 public final String name;  
  
 private Token() {  
 this((String)null);  
 }  
  
 private Token(String name) {  
 this.name = name;  
 }  
}