### Sample grammar for ATIS queries

### CS187

### (Project segment 4 distribution version)

## This grammar is intended to cover ATIS flight ID queries only. It

## assumes that such queries are a single noun phrase, perhaps

## preceded and followed by other text that can be ignored. (The

## nonterminals PREIGNORE and POSTIGNORE capture this text.)

##

## Conventions used in this grammar:

##

## The start nonterminal is S. Other nonterminal names are typically

## syntactic phrase types (e.g., NP, PP, Det, P) followed by a

## semantic category (e.g., FLIGHT, AIRLINE), separated by an underscore.

## The phrase type TERM is used for lexical lists for a particular semantic

## category.

##

## Augmentations for these grammar rules follow a standard convention for the

## types of the values associated with different nonterminals. We document these

## conventions here:

##

## Values: SQL query (as a string), e.g.,

## "SELECT flight.flight\_id FROM flight WHERE true"

##

## S

##

## Values: SQL condition (that is, appropriate for a WHERE clause body)

## for a relation `flight`, e.g.,

## "flight.airline\_code = 'UA'"

##

## NOM\_FLIGHT

## N\_FLIGHT

## PP

## PP\_\*

## ADJ

## ADJ\_\*

## TERM\_FLIGHT

## etc.

S -> NP\_FLIGHT :select

| PREIGNORE NP\_FLIGHT :lambda PRE, NPF : select(NPF)

| NP\_FLIGHT POSTIGNORE :lambda NPF, POST : select(NPF)

| PREIGNORE NP\_FLIGHT POSTIGNORE :lambda PRE, NPF, POST : select(NPF)

NP\_FLIGHT -> DET NOM\_FLIGHT :second

| NOM\_FLIGHT :first

NOM\_FLIGHT -> N\_FLIGHT :first

| ADJ NOM\_FLIGHT :conjoin

N\_FLIGHT -> TERM\_FLIGHT :first

| N\_FLIGHT PP :conjoin

TERM\_FLIGHT -> 'flights' :null\_condition

| 'flight'

| 'to' 'fly'

## Pre-noun (primarily adjectival) flight modifiers

ADJ -> ADJ\_AIRLINE :first

| ADJ\_DATE

| ADJ\_FLIGHTTYPE

| ADJ\_TIME

| ADJ\_PLACE

| ADJ\_PRICE

| ADJ\_CLASS

| ADJ\_FOOD

## Post-noun (prepositional phrase and relative clause) flight modifiers

PP -> PP\_AIRLINE :first

| PP\_DATE

| PP\_TIME

| PP\_PLACE

| PP\_PRICE

| PP\_CLASS

| PP\_FOOD

DET -> 'all' 'the' :ignore

| 'all'

| 'a'

| 'an'

| 'the'

| 'any'

| 'all' 'of' 'the'

| 'this'

| 'this' 'coming'

| 'next'

| TERM\_WEEKDAY "'s"

#### FLIGHT PROPERTIES ####

## Airline information

ADJ\_AIRLINE -> TERM\_AIRLINE :first

PP\_AIRLINE -> P\_AIRLINE TERM\_AIRLINE :second

P\_AIRLINE -> 'on' :ignore

| 'using'

| 'of'

| 'with'

TERM\_AIRLINE -> TERM\_AIRBRAND :airline\_name

| TERM\_AIRBRAND TERM\_AIRBRANDTYPE :lambda brand, type: airline\_name(brand)

TERM\_AIRBRAND -> 'continental' : constant('CO')

| 'american' : constant('AA')

| 'united' : constant('UA')

| 'northwest' : constant('NW')

| 'us' : constant('US')

| 'delta' : constant('DL')

| 'air' 'canada' : constant('AC')

| 'midwest' 'express' : constant('YX')

| 'twa' | 'trans' 'world' : constant('TW')

TERM\_AIRBRANDTYPE -> 'airline' : ignore

| 'airlines'

| 'air'

## Flight type information

ADJ\_FLIGHTTYPE -> ADJ\_FLIGHTTYPESIMPLE

| ADJ\_FLIGHTTYPESIMPLE 'and' ADJ\_FLIGHTTYPE

| ADJ\_FLIGHTTYPESIMPLE 'or' ADJ\_FLIGHTTYPE

ADJ\_FLIGHTTYPESIMPLE -> 'round' 'trip'

| 'round-trip'

| 'return'

| 'one' 'way'

| 'nonstop'

| 'direct'

| 'connecting'

## Other properties (cost, class, food)

ADJ\_PRICE -> 'cheapest'

| 'lowest' 'cost'

| 'least' 'expensive'

| 'inexpensive'

| 'cheap'

| 'expensive'

| 'highest' 'cost'

| 'most' 'expensive'

PP\_PRICE -> 'less' 'than' AMOUNT 'dollars'

| 'with' 'the' 'lowest' 'fare'

| 'the' 'cheapest' 'way' 'possible'

| 'with' 'the' 'highest' 'fare'

ADJ\_CLASS -> 'economy'

| 'thrift' 'economy'

| 'first' 'class'

| 'transcontinental'

| 'available'

| 'possible'

PP\_CLASS -> ADJ\_CLASS

ADJ\_FOOD -> NP\_FOOD

PP\_FOOD -> P\_FOOD NP\_FOOD

NP\_FOOD -> 'dinner'

| 'lunch'

| 'breakfast'

| 'a' 'meal'

P\_FOOD -> 'serving'

| 'with'

#### DATES ####

ADJ\_DATE -> TERM\_WEEKDAY

PP\_DATE -> P\_DATE NP\_DATE :second

| NP\_DATE :first

P\_DATE -> 'on' :ignore

| 'returning' 'on'

| 'of'

| 'for'

| 'next'

| 'the' 'next'

| 'in' 'the' 'next'

| 'of' 'next'

| 'leaving'

| 'departing'

| 'departing' 'on'

| 'which' 'leave'

| 'leaving' 'on'

| 'arriving'

| 'arriving' 'on'

| 'that' 'arrive' 'on'

| 'which' 'arrive' 'on'

| 'a' 'week' 'from'

NP\_DATE -> TERM\_WEEKDAY :first

| 'a' TERM\_WEEKDAY

| 'this' TERM\_WEEKDAY

| 'this' 'coming'

| TERM\_WEEKDAY ADJTIME

| NP\_MDYDATE

| 'saturdays'

| 'sundays'

| 'mondays'

| 'tuesdays'

| 'wednesdays'

| 'thursdays'

| 'fridays'

| 'day'

| 'week'

| 'today'

| 'tomorrow'

| 'the' 'day' 'after' 'tomorrow'

| 'weekdays'

NP\_MDYDATE -> TERM\_MONTH TERM\_DAY YEAR

| TERM\_MONTH TERM\_DAY

| 'the' TERM\_DAY 'of' TERM\_MONTH

| TERM\_DAY 'of' TERM\_MONTH

| 'the' TERM\_DAY

| TERM\_MONTH TERM\_DAY 'or' TERM\_DAY

| 'either' TERM\_MONTH TERM\_DAY 'or' TERM\_DAY

| 'either' 'the' TERM\_DAY 'or' 'the' TERM\_DAY 'of' TERM\_MONTH

| 'the' TERM\_DAY 'or' TERM\_DAY

| 'either' 'the' TERM\_DAY 'or' 'the' TERM\_DAY

| 'the' TERM\_DAY 'of' TERM\_MONTH 'or' 'the' TERM\_DAY 'of' TERM\_MONTH

| 'either' 'the' TERM\_DAY 'of' TERM\_MONTH 'or' 'the' TERM\_DAY 'of' TERM\_MONTH

TERM\_WEEKDAY -> 'saturday' : constant(weekday(\_RHS[0]))

| 'sunday'

| 'monday'

| 'tuesday'

| 'wednesday'

| 'thursday'

| 'friday'

| 'weekday'

TERM\_MONTH -> 'january' : constant(month\_name(\_RHS[0]))

| 'february'

| 'march'

| 'april'

| 'may'

| 'june'

| 'july'

| 'august'

| 'september'

| 'october'

| 'november'

| 'december'

TERM\_DAY -> 'first' : constant(1)

| 'second' : constant(2)

| 'third' : constant(3)

| 'fourth' : constant(4)

| 'fifth' : constant(5)

| 'sixth' : constant(6)

| 'seventh' : constant(7)

| 'eighth' : constant(8)

| 'ninth' : constant(9)

| 'tenth' : constant(10)

| 'eleventh' : constant(11)

| 'twelfth' : constant(12)

| 'thirteenth' : constant(13)

| 'fourteenth' : constant(14)

| 'fifteenth' : constant(15)

| 'sixteenth' : constant(16)

| 'seventeenth' : constant(17)

| 'eighteenth' : constant(18)

| 'nineteenth' : constant(19)

| 'twentieth' : constant(20)

| 'twenty-first' : constant(21)

| 'twenty' 'first' : constant(21)

| 'twenty-second' : constant(22)

| 'twenty' 'second' : constant(22)

| 'twenty-third' : constant(23)

| 'twenty' 'third' : constant(23)

| 'twenty-fourth' : constant(24)

| 'twenty' 'fourth' : constant(24)

| 'twenty-fifth' : constant(25)

| 'twenty' 'fifth' : constant(25)

| 'twenty-sixth' : constant(26)

| 'twenty' 'sixth' : constant(26)

| 'twenty-seventh' : constant(27)

| 'twenty' 'seventh' : constant(27)

| 'twenty-eighth' : constant(28)

| 'twenty' 'eighth' : constant(28)

| 'twenty-ninth' : constant(29)

| 'twenty' 'ninth' : constant(29)

| 'thirtieth' : constant(30)

| 'thirty-first' : constant(31)

| 'thirty' 'first' : constant(31)

| 'one' : constant(1)

| 'two' : constant(2)

| 'three' : constant(3)

| 'four' : constant(4)

| 'five' : constant(5)

| 'six' : constant(6)

| 'seven' : constant(7)

| 'eight' : constant(8)

| 'nine' : constant(9)

| 'ten' : constant(10)

| 'eleven' : constant(11)

| 'twelve' : constant(12)

| 'thirteen' : constant(13)

| 'fourteen' : constant(14)

| 'fifteen' : constant(15)

| 'sixteen' : constant(16)

| 'seventeen' : constant(17)

| 'eighteen' : constant(18)

| 'nineteen' : constant(19)

| 'twenty' : constant(20)

| 'twenty' 'one' : constant(21)

| 'twenty' 'two' : constant(22)

| 'twenty' 'three' : constant(23)

| 'twenty' 'four' : constant(24)

| 'twenty' 'five' : constant(25)

| 'twenty' 'six' : constant(26)

| 'twenty' 'seven' : constant(27)

| 'twenty' 'eight' : constant(28)

| 'twenty' 'nine' : constant(29)

| 'thirty' : constant(30)

| 'thirty' 'one' : constant(31)

TERM\_YEAR -> '1991' : constant(int(\_RHS[0]))

| '1992'

#### TIMES ####

PP\_TIME -> P\_TIME NP\_TIME :forward

P\_TIME -> 'that' 'arrive' 'before' :constant(arrive\_before)

| 'that' 'arrives' 'before'

| 'arriving' 'before'

| 'arrival' 'by'

| 'which' 'arrive' 'before'

| 'departing' 'before' :constant(depart\_before)

| 'before'

| 'that' 'leaves' 'before'

| 'by'

| 'that' 'return' 'around'

| 'that' 'gets' 'in' 'around'

| 'arriving' 'around'

| 'arriving' 'about'

| 'that' 'arrive' 'soon' 'after'

| 'around'

| 'arrives'

| 'arriving' 'at'

| 'arriving'

| 'leaving' 'at'

| 'at'

| 'leaving'

| 'in'

| 'departing' 'at'

| 'on'

| 'that' 'leaves' 'at'

| 'arriving' 'after'

| 'which' 'arrives' 'after'

| 'that' 'arrives' 'after'

| 'which' 'leave' 'after'

| 'leaving' 'after'

| 'after'

| 'departing' 'after'

| 'departing' 'in'

| 'that' 'depart' 'after'

| 'that' 'leaves' 'in'

NP\_TIME -> TERM\_TIME : first

| 'afternoons' : constant(1200)

| 'the' 'afternoon'

| 'the' 'late' 'afternoon' : constant(1600)

| 'evenings' : constant(1800)

| 'the' 'evening'

| 'mornings' : constant(800)

| 'the' 'morning'

| 'the' 'early' 'am'

| 'as' 'early' 'as' 'possible'

| 'earliest' 'possible' 'time'

| 'the' 'day' : constant(1200)

| 'as' 'soon' 'thereafter' 'as' 'possible'

| UNK

ADJ\_TIME -> 'lunch' 'time' : constant(depart\_around(1200))

| 'evening' : constant(depart\_around(1800))

| 'dinnertime'

| 'late'

| 'night' : constant(depart\_around(2000))

| 'latest' 'possible'

| 'latest'

| 'tonight'

| 'morning' : constant(depart\_around(800))

| 'early'

| 'earliest' 'possible'

| 'earliest'

| 'afternoon' : constant(depart\_around(1600))

| SIMPLETIME : first

TERM\_TIME -> 'one' | '1' : constant(100)

| 'two' | '2' : constant(200)

| 'three' | '3' : constant(300)

| 'four' | '4' : constant(400)

| 'five' | '5' : constant(500)

| 'six' | '6' : constant(600)

| 'seven' | '7' : constant(700)

| 'eight' | '8' : constant(800)

| 'nine' | '9' : constant(900)

| 'ten' | '10' : constant(1000)

| 'eleven' | '11' : constant(1100)

| 'twelve' | '12' | 'noon' : constant(1200)

| 'midnight' : constant(0)

| TERM\_TIME TERM\_TIMEMOD : lambda Time, Mod: Time + Mod

TERM\_TIMEMOD -> 'am' : constant(0)

| 'oclock'

| 'o' "'clock"

| 'o' 'clock'

| 'noon'

| 'pm' : constant(1200)

| 'midnight'

#### PLACES ####

ADJ\_PLACE -> N\_PLACE

N\_PLACE -> TERM\_PLACE :first

| TERM\_AIRPORT

PP\_PLACE -> P\_PLACE N\_PLACE :forward

| 'between' N\_PLACE 'and' N\_PLACE :lambda a,b: conjoin(from\_place(a), to\_place(b))

P\_PLACE -> 'to' :constant(to\_place)

| 'that' 'arrive' 'at'

| 'that' 'arrives' 'in'

| 'coming' 'back' 'to'

| 'that' 'go' 'to'

| 'and' 'then' 'to'

| 'arriving' 'in'

| 'and' 'arriving' 'in'

| 'and' 'arrive' 'in'

| 'to' 'arrive' 'in'

| 'arrive' 'in'

| 'going' 'to'

| 'into'

| 'for'

| 'with' 'the' 'destination' 'city' 'of'

| 'arriving'

| 'goes' 'to'

| 'flying' 'into'

| 'goes' 'on' 'to'

| 'reaching'

| 'in'

| 'and' 'then'

| 'arriving' 'to'

| 'from' :constant(from\_place)

| 'leaving'

| 'return' 'from'

| 'leaving' 'from'

| 'departing' 'from'

| 'departing'

| 'go' 'from'

| 'going' 'from'

| 'back' 'from'

| 'that' 'goes' 'from'

| 'that' 'departs'

| 'which' 'leaves' 'from'

| 'which' 'leave'

| 'that' 'leave'

| 'originating' 'in'

| 'leave'

| 'out' 'of'

| 'leaves' 'from'

| 'to' 'get' 'from'

| 'via'

| 'with' 'a' 'stopover' 'in'

| 'with' 'a' 'layover' 'in'

| 'with' 'a' 'stopover' 'at'

| 'and' 'a' 'stopover' 'in'

| 'stop' 'in'

| 'stopping' 'in'

| 'make' 'a' 'stop' 'in'

| 'with' 'a' 'stop' 'in'

| 'with' 'one' 'stop' 'in'

| 'go' 'through'

| 'which' 'go' 'through'

| 'makes' 'a' 'stopover' 'in'

| 'that' 'stops' 'in'

| 'that' 'stops' 'over' 'in'

| 'by' 'way' 'of'

| 'connecting' 'through'

| 'that' 'will' 'stop' 'in'

| 'which' 'connects' 'in'

| 'arriving' 'and' 'departing' 'at'

## Place terms includes city and airport names. The provided semantic augmentations

## return a SQL query that generates a list of airport codes for the place. For

## airport names, like "logan", it generates a query like

##

## SELECT airport.airport\_code

## FROM airport

## WHERE airport.airport\_name = "LOGAN INTERNATIONAL"

##

## For city names, like "boston", it generates a query like

##

## SELECT airport\_service.airport\_code

## FROM airport\_service

## WHERE airport\_service.city\_code IN

## (SELECT city.city\_code

## FROM city

## WHERE city.city\_name = "BOSTON")

##

## Note the use of the appropriate field values found in the ATIS database, which are

## quoted uppercase strings like "LOGAN INTERNATIONAL" or "BOSTON"

TERM\_AIRPORT -> TERM\_AIRPORT 'airport' : first

| TERM\_AIRPORT 'airports'

| 'baltimore/washington' 'international' : constant(airports\_from\_airport\_name(' '.join(\_RHS)))

| 'burbank'

| 'buttonville'

| 'charlotte/douglas' 'international'

| 'cincinnati/northern' 'kentucky' 'intl.'

| 'dallas/fort' 'worth' 'international'

| 'detroit' 'city'

| 'dorval' 'international'

| 'dulles' 'international'

| 'general' 'mitchell' 'international'

| 'greater' 'pittsburgh' 'international'

| 'hobby'

| 'hopkins' 'international'

| 'houston' 'intercontinental'

| 'indianapolis' 'international'

| 'john' 'f.' 'kennedy' 'international'

| 'kansas' 'city' 'international'

| 'la' 'guardia'

| 'lambert' 'st.' 'louis' 'international'

| 'lester' 'b.' 'pearson' 'international'

| 'lindbergh' 'field/san' 'diego' 'international'

| 'logan' 'international'

| 'long' 'beach' 'municipal'

| 'los' 'angeles' 'international'

| 'love' 'field'

| 'mccarran' 'international'

| 'memphis' 'international'

| 'metropolitan' 'oakland' 'international'

| 'metropolitan' 'wayne' 'county'

| 'miami' 'international'

| 'midway'

| 'minneapolis/st.' 'paul' 'international'

| 'mirabel'

| 'nashville' 'international'

| 'newark' 'international'

| "o'hare" 'international'

| 'ontario' 'international'

| 'orlando' 'international'

| 'philadelphia' 'international'

| 'port' 'columbus' 'international'

| 'salt' 'lake' 'city' 'international'

| 'san' 'francisco' 'international'

| 'san' 'jose' 'international'

| 'seattle/tacoma' 'international'

| 'sky' 'harbor' 'international'

| 'st.' 'petersburg/clearwater' 'international'

| 'stapleton' 'international'

| 'tampa' 'international'

| 'toronto' 'island'

| 'washington' 'national'

| 'westchester' 'county'

| 'william' 'b.' 'hartsfield' 'atlanta' 'intl.'

## airport name synonyms

| 'bwi' : constant(airports\_from\_airport\_name('baltimore/washington international'))

| 'douglas' : constant(airports\_from\_airport\_name('charlotte/douglas international'))

| 'dfw' : constant(airports\_from\_airport\_name('dallas/fort worth international'))

| 'dorval' : constant(airports\_from\_airport\_name('dorval international'))

| 'dulles' : constant(airports\_from\_airport\_name('dulles international'))

| 'general mitchell' : constant(airports\_from\_airport\_name('general mitchell international'))

| 'hopkins' : constant(airports\_from\_airport\_name('hopkins international'))

| 'jfk' : constant(airports\_from\_airport\_name('john f. kennedy international'))

| 'john' 'f.' 'kennedy'

| 'kennedy'

| 'laguardia' : constant(airports\_from\_airport\_name('la guardia'))

| 'lambert' : constant(airports\_from\_airport\_name('lambert st. louis international'))

| 'pearson' : constant(airports\_from\_airport\_name('lester b. pearson international'))

| 'lindbergh' : constant(airports\_from\_airport\_name('lindbergh field/san diego international'))

| 'logan' : constant(airports\_from\_airport\_name('logan international'))

| 'mccarran' : constant(airports\_from\_airport\_name('mccarran international'))

TERM\_PLACE -> 'atlanta' : constant(airports\_from\_city(' '.join(\_RHS)))

| 'austin'

| 'baltimore'

| 'boston'

| 'burbank'

| 'charlotte'

| 'chicago'

| 'cincinnati'

| 'cleveland'

| 'columbus'

| 'dallas'

| 'denver'

| 'detroit'

| 'houston'

| 'indianapolis'

| 'kansas' 'city'

| 'las' 'vegas'

| 'long' 'beach'

| 'los' 'angeles'

| 'memphis'

| 'miami'

| 'milwaukee'

| 'minneapolis'

| 'montreal'

| 'nashville'

| 'new' 'york'

| 'newark'

| 'oakland'

| 'ontario'

| 'orlando'

| 'philadelphia'

| 'phoenix'

| 'pittsburgh'

| 'salt' 'lake' 'city'

| 'san' 'diego'

| 'san' 'francisco'

| 'san' 'jose'

| 'seattle'

| 'st.' 'louis'

| 'st.' 'paul'

| 'st.' 'petersburg'

| 'tacoma'

| 'tampa'

| 'toronto'

| 'washington'

| 'dc'

| 'westchester' 'county'

## city name synonyms

| 'cleveland' 'ohio' : constant(airports\_from\_city('cleveland'))

| 'dallas' 'fort' 'worth' : constant(airports\_from\_city('dallas'))

| 'fort' 'worth' : constant(airports\_from\_city('dallas'))

| 'denver' 'colorado' : constant(airports\_from\_city('denver'))

| 'montreal' 'quebec' : constant(airports\_from\_city('montreal'))

| 'montreal' 'canada' : constant(airports\_from\_city('montreal'))

| 'new' 'york' 'city' : constant(airports\_from\_city('new york'))

| 'newark' 'new' 'jersey' : constant(airports\_from\_city('newark'))

| 'oakland' 'california' : constant(airports\_from\_city('oakland'))

| 'orlando' 'florida' : constant(airports\_from\_city('new york'))

| 'philly' : constant(airports\_from\_city('philadelphia'))

| 'san' 'diego' 'california' : constant(airports\_from\_city('san diego'))

| 'tacoma' 'washington' : constant(airports\_from\_city('tacoma'))

| 'washington' 'dc' : constant(airports\_from\_city('washington'))

| 'westchester' : constant(airports\_from\_city('westchester'))

#### IGNORABLE ####

PREIGNORE -> PREIGNORESYMBOL : ignore

| PREIGNORESYMBOL PREIGNORE

PREIGNORESYMBOL -> 'me' : ignore

| 'show'

| 'now'

| 'only'

| 'can'

| 'you'

| 'the'

| 'itinerary'

| 'of'

| 'also'

| 'a'

| 'list'

| 'could'

| 'give'

| 'which'

| 'what'

| 'is'

| "what's"

| 'are'

| 'my'

| 'choices'

| 'for'

| 'i'

| 'would'

| 'like'

| "i'd"

| 'to'

| 'see'

| 'have'

| 'make'

| 'book'

| 'find'

| 'information'

| 'on'

| 'know'

| 'some'

| 'hello'

| 'yes'

| 'please'

| 'repeat'

| 'do'

| 'have'

| 'there'

| 'need'

| 'hi'

| 'get'

| 'may'

| 'listing'

| 'listings'

| 'travel'

| 'arrangements'

| 'okay'

| 'want'

| 'tell'

| 'about'

| 'how'

| 'would'

| 'be'

| 'able'

| 'put'

| 'requesting'

| "i'm"

| 'looking'

| 'display'

POSTIGNORE -> POSTIGNORESYMBOL : ignore

| POSTIGNORESYMBOL POSTIGNORE

POSTIGNORESYMBOL -> 'please' : ignore

| 'there'

| 'are'

| 'currently'

| 'do'

| 'you'

| 'have'

| 'fares'

| 'information'

| 'i'

| 'want'

| 'would'

| 'like'

| 'the'

| 'flight'

| 'be'

| 'go'

| 'departures'

| 'is'

| 'such'

| 'a'

| 'that'

| 'serves'

| 'both'

| 'and'

| 'along'

| 'with'

| 'can'

| 'get'

| "i'd"

| 'traveling'

| 'for'

| 'me'

| UNK

| '.'

| '?'