A Appendix: ILP Model Details

To build the ILP model, we first obtain the questions terms (qterm) from the question by chunking the question using an in-house chunker based on the postagger from FACTORIE. [14] We ignore chunks that only contain stop-words.

Variables
The ILP model has an active vertex variable for each qterm (x_q), tuple (x_t), tuple field (x_f) and question choice (x_a). Table 4 describes the coefficients of these active variables. For example, the coefficient of each qterm is a constant value (0.8) scaled by three boosts. The idf boost, $idfB$ for a qterm, x is calculated as $\log(1 + (|T_{qa}| + |T'_{qa}|)/n_x)$ where n_x is the number of tuples in $T_{qa} \cup T'_{qa}$ containing x. The science term boost, $scienceB$ (set to 2.0) boosts coefficients of qterms that are valid science terms based on a list of 9K terms. The location boost, $locB$ of a qterm at index i in the question is given by $i/tok(q)$ (where i=1 for the first term). Finally the objective function of our ILP model can be written as:

$$\sum_{q \in \text{qterms}} c_q x_q + \sum_{t \in \text{tuples}} c_t x_t + \sum_{e \in \text{edges}} c_e x_e$$

Description	Var.	Coefficient (c)
Qterm	x_q	0.8·idfB·scienceB·locB
Tuple	x_t	-1 + jaccardScore(t, qa)
Tuple Field	x_f	0
Choice	x_a	0

Table 4: Coefficients for active variables.

Constraints Apart from the constraints described in Table 1, we also use the *which-term* boosting constraints defined by TABLEILP (Eqns. 44 and 45 in Table 13 (Khashabi et al., 2016)). As described in Section B, we create a tuple from table rows by setting pairs of cells as the subject and object of a tuple. For these tuples, apart from requiring the subject to be active, we also require the object of the tuple to be active. This would be equivalent to requiring at least two cells of a table row to be active.

[14]http://factorie.cs.umass.edu/

B Experiment Details

We use the SCIP ILP optimization engine (Achterberg, 2009) to optimize our ILP model. To get the score for each answer choice a_i, we force the active variable for that choice x_{a_i} to be one and use the objective function value of the ILP model as the score. For each question, a solver gets a score of 1 if it chooses the correct answer and $1/k$ if it reports a k-way tie that includes the correct answer. For evaluations, we use a 2-core 2.5 GHz Amazon EC2 linux machine with 16 GB RAM. To evaluate TABLEILP and TUPLEINF on curated tables and tuples, we converted them into the expected format of each solver as follows.

B.1 Using curated tables with TUPLEINF

For each question, we select the 7 best matching tables using the tf-idf score of the table w.r.t. the question tokens and top 20 rows from each table using the Jaccard similarity of the row with the question. (same as Khashabi et al. (2016)). We then convert the table rows into the tuple structure using the relations defined by TABLEILP. For every pair of cells connected by a relation, we create a tuple with the two cells as the subject and primary object with the relation as the predicate. The other cells of the table are used as additional objects to provide context to the solver. We pick top-scoring 50 tuples using the Jaccard score.

B.2 Using Open IE tuples with TABLEILP

We create an additional table, T_O in TABLEILP using all the tuples in our KB, T. Since TABLEILP uses fixed-length $(subject; predicate; object)$ triples, we need to map tuples with multiple objects to this format. For each object, O_i in the input Open IE tuple $(S; P; O_1; O_2 \ldots) \in T$, we add a triple $(S; P; O_i)$ to the table, T_O.