Table 7

SQL Data Manipulation Language (DML) Minimum Construct Set

DML Constructs	**Justification**
Selecting data	The means by which to extract data from a database management system (DBMS).
• Basic predicates: simple operators, [NOT] IN, [NOT] NULL, and DISTINCT	These predicates are employed for the most basic of SQL transactions, from basic querying to profiling to analytics.
• Aggregate functions	Aggregate functions such as average, count, max, min, and sum are readily used in reporting and basic data analysis and profiling.
• Advanced predicates: regular expressions, ranges, GROUP BY, HAVING, ORDER BY, and LIMIT	Many attributes are text-based and therefore require some form of regular expression for querying. Additionally, GROUP BY and HAVING are essential for aggregating and limiting subsets of data. ORDER BY is used for sorting, and LIMIT is used to constrain the number of results displayed. Each of these functions is necessary for database interrogation.
• Parentheses and SQL order of execution	Many questions in healthcare contain two or more sets of conditions, for example, patients greater than 65 with at least one disease from set A and one disease from set B. If the user does not have a proper understanding of the order of SQL execution and the effects of parentheses, the DBMS will incorrectly interpret the query, returning logically incorrect (yet syntactically correct) results.
• Joins: inner, natural, and Cartesian	One of the most basic SQL design patterns, a join allows a relation to be combined with another in a meaningful way. Inner joins are those most commonly associated with the term *join*. That is, the results contain only those tuples (records) existing in both sets. Natural joins achieve the same goal using an implicit join predicate. That is, a natural join relies on attribute names and data types from the relations being joined to determine how to combine them. Cartesian joins return the product of the relations (i.e., each tuple in one relation is matched with every tuple in the other). While Cartesian joins have legitimate uses, novice students should be aware of their existence because the result of a Cartesian join is effectively what they would see if they failed to include a join predicate for an inner join. Thus, exposing students to Cartesian joins is necessary for error recognition and recovery.
• Noncorrelated subqueries	Subqueries allow the results of one query to be used as a predicate value in another. For example, to determine the patient with the greatest weight, the maximum weight must first be known (the subquery). That value is then used to locate the encounter(s) with that recorded weight (outer query). This example is a noncorrelated subquery, in which the results of the subquery do not depend on the outer query.
• Views	Declaring a view is a Data Definition Language (DDL) command, but the query itself is any SELECT statement. From a DML perspective, views simplify employing queries and enhance programming productivity while maintaining access to current information (not true for materialized views). In an introductory setting, many students may be sufficiently challenged by basic SELECT statements, limiting the need to focus on views. At a minimum, student should have an understanding of what views accomplish and how to create them.
Inserting: single tuples, from SELECT	The means by which to populate tables. Data can be added one item at a time or from the results of a query. Although the former method is useful, the latter is commonly used for constructing secondary data sets, copying tables or results, and de-normalization.
Updating and deleting	The means by which to manage existing tuples.