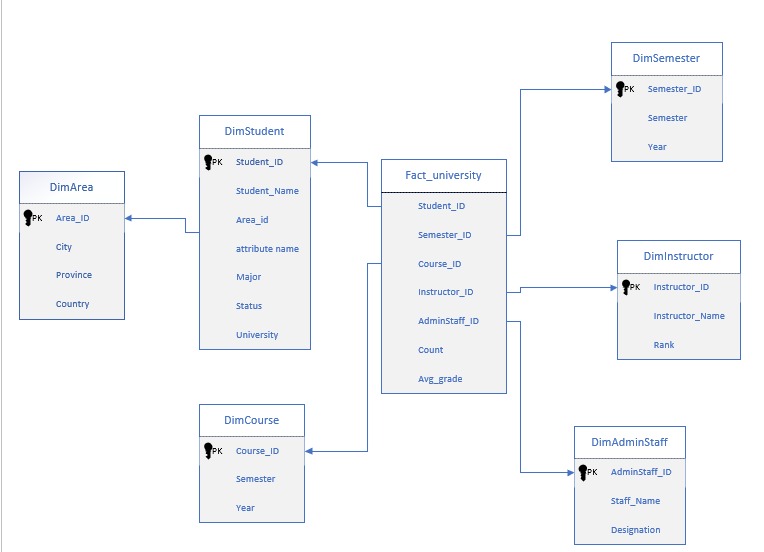
**ASSIGNMENT NO 3**

**DATAWARE HOUSING**

**CS-452**

**Question 1:**



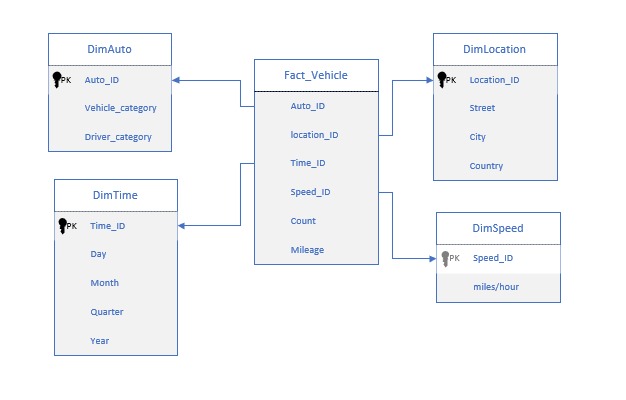
1. The specific OLAP operations to be performed are:

* Roll-up on course from course id to department.
* Roll-up on semester from semester id to year
* Slice for course= “CS” .

1. This cube will contain 5^5 = 3125 cuboids.

**Question 2:**

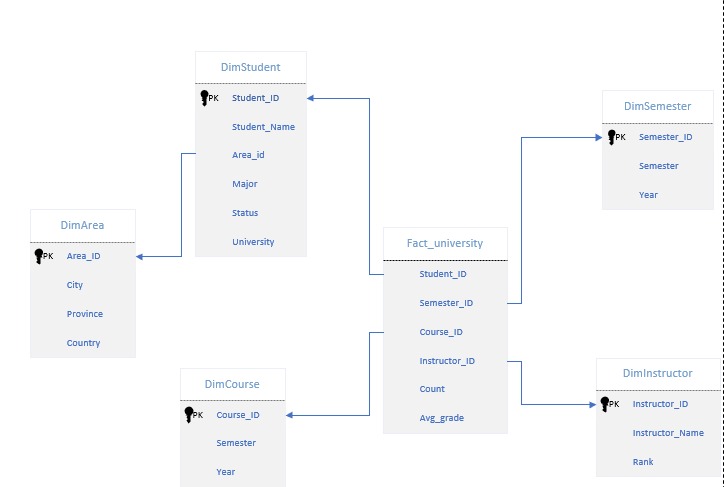
1. To design a data warehouse for the analysis of moving vehicles, consider vehicle as a fact table that points to four dimensions: auto, time, location, and speed.



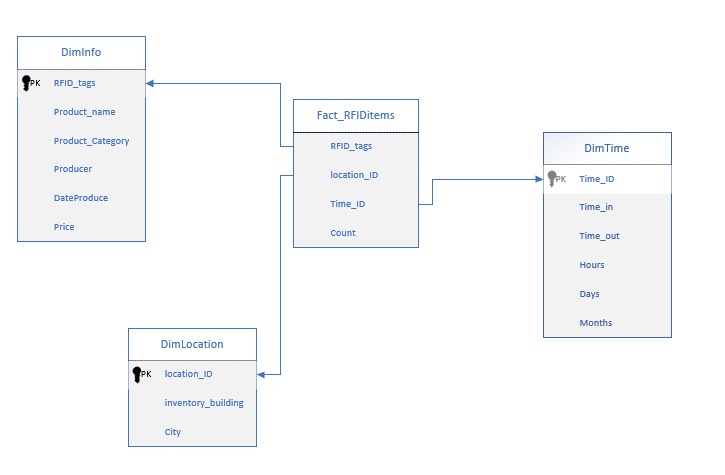
**b)** To handle the noise in data, we first need to do data cleaning. Missing values may be filled or dropped entirely, depending on the tolerance of the system. Then we can use some data smoothing techniques to remove noisy data points.

**c)** Sparse data may exist in the data warehouse, and analyzing it can be unreliable due to the potential impact of outliers. Efficient methods are available to handle sparse data, such as using existing data to estimate missing values. This approach is commonly employed in machine learning. For example, if a speed value is missing, it can be inferred based on the location or time using previously recorded speeds for that street and time. By maintaining query semantics, it is possible to reduce the dimensionality of the data cube. For instance, if a cell was sparse for a specific hour, it can be generalized to represent a day, resulting in populated values within that cell.

**d)** Leveraging the data warehouse, it is possible to retrieve information for vehicles within the same category as well as drivers within the same category. Through OLAP operations like drill and dice, it is possible to query the speed of a location at a specific time (typically at the hour level) and use it as the weight for the corresponding street on the city graph. By calculating the weighted graph, it becomes feasible to determine the fastest route for a driver using algorithms like Dijkstra's. It may be necessary to update the weights on an hourly basis. This approach does not consider the direction of streets and can be further enhanced by incorporating the information into a directed graph. Based on this graph, the fastest route can be calculated.

****

**Question 3:**

a) 

**b)** Each reader provides sets of the form (RFID; location; time). When an item stays at the same location for a period, multiple sets will be generated. We can group these sets into one of the forms (RFID; location; time in; time out).

**c)** One can use the assumption that many RFID objects stay or move together, especially at the early stage of distribution, or use the historically most likely path for a given item, to infer or interpolate the miss and error reading.

**d)** Compute an aggregate measure on the tags that travel through a set of locations and that match a selection criterion on path independent dimensions.

**e)** For this case, after we obtain the RFID of the milk, we can directly use traditional OLAP operations to get the shipping and storage time efficiently.

**Question 4:**

**A picture containing text, diagram, line, screenshot

Description automatically generated**

**Question 5:**

**(a)**

Dimension: Call Program

Hierarchy: Call Program (all)

Measure: Amount

OLAP Operation: Summarize by Call Program, filter for the year 2012, and calculate the sum of Amount.

**(b)**

Dimension: Caller Customer, Time

Hierarchy: Caller Customer > Location > City (including Brussels), Time (all)

Measure: Duration

OLAP Operation: Summarize by Caller Customer City (Brussels), filter for the year 2012, and calculate the sum of Duration.

**(c)**

Dimensions: Caller Customer, Callee Customer, Time

Hierarchies: Caller Customer > Location > City (including Brussels), Callee Customer > Location > City (including Antwerp), Time > Day > Weekend

Measure: Number of Calls

OLAP Operation: Summarize by Caller Customer City (Brussels), Callee Customer City (Antwerp), filter for the year 2012 and Weekend, and calculate the sum of Number of Calls.

**(d)**

Dimensions: Caller Customer, Time, Call Type

Hierarchies: Caller Customer > Country (including Belgium), Time (all), Call Type > International

Measure: Duration

OLAP Operation: Summarize by Caller Customer Country (Belgium), filter for the year 2012 and international calls, and calculate the sum of Duration.

**(e)**

Dimensions: Caller Customer, Call Program

Hierarchies: Caller Customer > Location > City (including Brussels), Call Program > Corporate

Measure: Amount

OLAP Operation: Summarize by Caller Customer City (Brussels), Call Program Corporate, filter for the year 2012, and calculate the sum of Amount.

**Question 6:**

a) The four stages of Data Warehouse are as follow's:

**1.Sources:**

* External Data
* Text Data

**2.Staging**

* ETL PROCESS
* Staging Area

**3.Data Warehouse:**

* DATA MARTS
* ODS
* ERP

**4.Presentation:**

* MARKETING
* SALES
* DATA MINING
* EXECUTIVE REPORTS

**Question 7:**

**a)** Dimensions: Train, Departure Station, Departure Time, Trip

Hierarchies: Train > name (including Alstom trains), Departure station > location > Country (including French and Belgian) > Departure Time > day > week > month > year, trip (all)

Measure: number of kilometers

OLAP Operation: Summarize by Train (Alstom trains) and Departure station (French and Belgian), filter for the year 2012, and calculate the sum of number of kilometers.

**b)** Dimensions: Trip, Departure Station, Departure Time, Arrival Station

Hierarchies: Trip > international, Departure station > location > Country, Departure Time > day > week > month > year, Arrival Station > Location > country

Measure: Duration

OLAP Operation: Summarize by Trip, filter for the year 2012, Departure Country <> Arrival Country, and calculate the sum of Duration.

**c)** Dimensions: Departure Station, Arrival Station, Trip, Departure Time

Hierarchies: Departure Station > location > City (including Paris), Arrival Station > location > City (including Paris), Trip (all), Departure Time > day > week > Month (July 2012)

Measure: Number of Trips

OLAP Operation: Summarize by Departure City (Paris) or Arrival City (Paris), filter for July 2012, and calculate the sum of Number of Trips.

**d)** Dimensions: Departure Station, Arrival Station, Trip, Departure Time, Train

Hierarchies: Departure Station > location > Country (including Belgium), Arrival Station > location > Country (including Belgium), Trip (all), Departure Time > day > week > month > Year (2012), Train (all)

Measure: Duration

OLAP Operation: Summarize by Departure Country (Belgium) and Arrival Country (Belgium), filter for the year 2012, and calculate the average of Duration for all train segments.

**e)** Dimensions: Trip, Train

Hierarchies: Trip (all), Train(all)

Measure: Number of Passengers

OLAP Operation: Summarize by Trip and Train, and calculate the average of Number of Passengers.

**Question 8:**

**a)** Dimensions: Department, Professor, Time

Hierarchies: Department, Professor, Time > Academic Year (2012-2013)

Measures: Number of Hours

OLAP Operation: Summarize by Department, filter for the academic year 2012-2013, and calculate the sum of Number of Hours.

**b)** Dimensions: Department, Professor, Funding Agency, Project, Time

Hierarchies: Department, Professor, Funding Agency, Project, Time > Year (2012)

Measures: Amount

OLAP Operation: Summarize by Department, filter for the calendar year 2012, and calculate the sum of Amount.

**c)** Dimensions: Department, Professor, Project, Time

Hierarchies: Department, Professor, Project, Time > Year (2012)

Measures: Count of Professors

OLAP Operation: Summarize by Department, filter for the calendar year 2012, and calculate the count of unique Professors.

**d)** Dimensions: Professor, Department, Course, Time

Hierarchies: Professor, Department, Course, Time > Academic Year (2012-2013)

Measures: Count of Courses

OLAP Operation: Summarize by Professor, filter for the academic year 2012-2013, and calculate the count of unique Courses.

**e)** Dimensions: Department, Professor, Funding Agency, Project, Time

Hierarchies: Department, Funding Agency, Project, Time > Year (2012)

Measures: Count of Projects

OLAP Operation: Summarize by Department and Funding Agency, filter for the year 2012, and calculate the count of unique Projects.

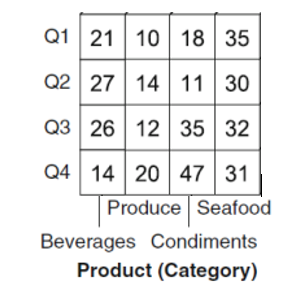
**Question 9:**

1. **Drill down on Time**

**A picture containing text, number, diagram, crossword puzzle

Description automatically generated**

1. **Slice on Paris**

****

1. **Dice on Customer city= Paris and Time = Q1 and Q2**

**A picture containing text, font, diagram, number

Description automatically generated**

1. **Dice on Customer = Paris and Time = Q1**

**A picture containing text, font, diagram, number

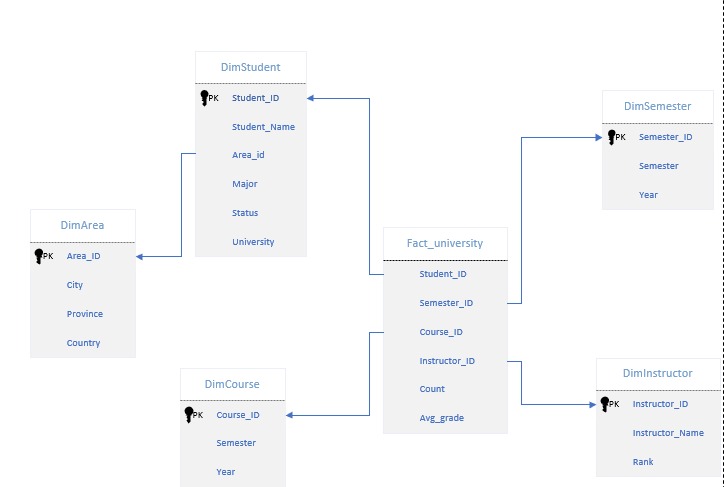
Description automatically generated**

1. **Dice on Customer = Paris and Time = Q1**

**A picture containing diagram, number, line, font

Description automatically generated**

**Question 10:**

****

1. The specific OLAP operations to be performed are:

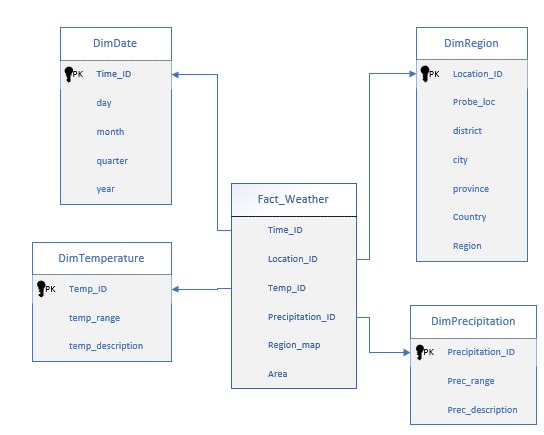
* Roll-up on course from course id to department.
* Roll-up on semester from semester id to year
* Slice for course=“CS” .

**Question 11:**

Here are some common properties of aggregation operators:

1. **Closure:** Closure refers to the property of an aggregation operator where the result obtained by applying the operator to a set of values is still within the same set. It means that the aggregation operation should produce a meaningful and consistent result that is compatible with the data being aggregated.
2. **Associativity:** Associativity relates to the characteristic of an aggregation operator where the grouping of elements does not impact the final result. In other words, regardless of how you group the elements, the outcome of applying an associative operator should remain the same. For example, the order of computation should not affect the result when aggregating three numbers: (a⋆b)⋆c should yield the same result as a⋆(b⋆c).
3. **Commutativity:** Commutativity implies that the order in which elements are combined using an aggregation operator does not affect the outcome. If an operator is commutative, aggregating two values (a and b) will yield the same result regardless of whether you compute a⋆b or b⋆a.
4. **Idempotence:** Idempotence refers to the property of an aggregation operator where applying it multiple times to the same value or set of values produces the same result as applying it only once. If an operator is idempotent, aggregating a value with itself multiple times will yield the same outcome as aggregating it just once.
5. **Monotonicity:** Monotonicity denotes the characteristic of an aggregation operator where the result follows the same trend as the input values. If an operator is monotonic, increasing or decreasing the input values will cause the result of the aggregation operator to exhibit the same pattern. Adding or removing values from the set being aggregated will not alter the overall result in terms of ordering.
6. **Null Handling:** Aggregation operators should handle null or missing values appropriately. Depending on the operator, null values may be ignored, treated as zero, or excluded from the aggregation process. Proper handling of null values ensures accurate and consistent aggregation results.
7. **Domain and Range:** Aggregation operators have specific domains, which define the set of values that can be aggregated, and ranges, which determine the possible outcomes or summaries generated by the operator. The domain specifies the valid input values, while the range defines the potential output values produced by the aggregation operation.

**Question 12:**



**Question 13:**

1. The three categories of measures are distributive, algebraic, and holistic.
2. The function variance is algebraic. If the cube is partitioned into many chunks, the variance can be computed as follows: Read in the chunks one by one, keeping track of the accumulated (1) number of tuples, (2) sum of (xi) 2 , and (3) sum of xi . After reading all the chunks, compute the average of xi ’s as the sum of xi divided by the total number of tuples.
3. each cuboid, use 10 units to register the top 10 sales found so far. Read the data in each cubiod once. If the sales amount in a tuple is greater than an existing one in the top-10 list, insert the new sales amount from the new tuple into the list, and discard the smallest one in the list. The computation of a higher-level cuboid can be performed similarly by propagation of the top-10 cells of its corresponding lower-level cuboids.

**Question 14:**

1. To support user preferences efficiently, a recommended approach is to utilize partial materialization in the data cube structure. By selectively computing specific cuboids instead of the entire set, storage space can be minimized while still ensuring fast response times and avoiding unnecessary calculations.
2. In scenarios where users need to drill through the cube for only a few dimensions, it can be beneficial to compute the necessary cuboids on demand. Since this feature may not be frequently used, the time required to compute aggregates for those specific dimensions on the fly should be acceptable to users.

**Question 15:**

5^12 = 2.4x10^8

**Question 16:**

3x6x6 = 108 cuboids

**Question 17:**

3x4x5 = 60 cuboids

**Question 18:**

1. Index structures are data structures used in database management systems to improve data access efficiency. They enable quick and targeted retrieval of data by optimizing query performance. One type of index is bitmap indexing, which represents the presence or absence of a value for each row in a table using a bitmap. Bitmap indexes are efficient for columns with a small number of distinct values and support fast bitwise operations for querying. Another type is join indexing, which registers the joinable rows of two relations in a relational database. Join indexing helps maintain the relationship between a foreign key and its matching primary keys.
2. When considering data cube materialization options for a base cuboid, there are three choices. The first choice is to not materialize any "non-base" cuboids, which leads to computing expensive multidimensional aggregates on-the-fly and can result in slow performance. The second choice is full materialization, where all cuboids are precomputed. This requires a significant amount of memory space to store all the precomputed cuboids. The third choice is partial materialization, where a proper subset of cuboids is selectively computed. This subset can be determined based on user-specified criteria, such as a threshold for tuple count. Partial materialization strikes a balance between storage space and response time, allowing for efficient querying while managing storage requirements.

**Question 19:**

**a)** Bitmap indexes represent the presence or absence of a value for each row in a table using a bitmap. Each bit in the bitmap corresponds to a row in the table, and its value indicates whether the associated value is present or not. Bitmap indexes are efficient for low cardinality columns (columns with a small number of distinct values) and support fast bitwise operations for querying.

**b)** item bitmap index table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| RID | H | C | P | S |
| R1 | 1 | 0 | 0 | 0 |
| R2 | 0 | 1 | 0 | 0 |
| R3 | 0 | 0 | 1 | 0 |
| R4 | 0 | 0 | 0 | 1 |
| R5 | 1 | 0 | 0 | 0 |
| R6 | 0 | 1 | 0 | 0 |
| R7 | 0 | 0 | 1 | 0 |
| R8 | 0 | 0 | 0 | 1 |

1. City bitmap

|  |  |  |
| --- | --- | --- |
| RID | V | T |
| R1 | 1 | 0 |
| R2 | 1 | 0 |
| R3 | 1 | 0 |
| R4 | 1 | 0 |
| R5 | 0 | 1 |
| R6 | 0 | 1 |
| R7 | 0 | 1 |
| R8 | 0 | 1 |

**Question 20:**

1. join indexing registers the joinable rows of two relations from a relational database. Join indexing is especially useful for maintaining the relationship between a foreign key and its matching primary keys, from the joinable relation. For example, if two relations R(RID, A) and S(B, SID) join on the attributes A and B, then the join index record contains the pair (RID, SID), where RID and SID are record identifiers from the R and S relations, respectively. Hence, the join index records can identify joinable tuples without performing costly join operations.
2. **(i) sales/location**

|  |  |
| --- | --- |
| **location** | **Sales\_key** |
| Main Street | T57 |
| Main Street | T238 |
| Main Street | T884 |

**(ii) item/sales**

|  |  |
| --- | --- |
| **Item** | **Sales\_key** |
| Sony-TV | T57 |
| Sony-TV | T459 |

**(iii) location and item to sales**

|  |  |  |
| --- | --- | --- |
| **Location** | **item** | **Sales\_key** |
| Main Street | Sony-TV | T57 |