Database Systems for Acceptance Sampling

Introduction: Database Systems

A database system is a triple of a database ***DB***, a set of application programs ***AP*** and a database management system *DBMS*. A database ***DB*** over a database schema S consists of a finite set $\boldsymbol{R}$ of relations (tables) $r \subseteq \mathrm{dom}(\boldsymbol{A})$ and a set of integrity constraints, $\boldsymbol{\Sigma}$, where $\mathrm{dom}(\boldsymbol{A})$ is the value set defined by the cross-product of the ranges of all attributes $A \in \boldsymbol{A}$. The attributes have specific data types like *int, real, string, Boolean*, and so on. The application programs support planning and management of batch inspection. The integrity constraints assure technical, statistical, and semantic coherency of the data stored in any $r \in \boldsymbol{R}$ at any time [1].

Database Systems for Acceptance Sampling

With respect to the granularity of data in ***DB*** and the objective of data retrieval we differentiate between on-line transaction processing (OLTP) and on-line analytical processing (OLAP) databases. OLTP databases used for "on-line transaction processing" have been in use since 1970 and are specially tuned to manage operative mass data from the shop floor [2]. They are used for on-line control like **acceptance sampling**, monitoring or **process control**, and in off-line control, for instance, for storing data of designed experiments. A typical query of **quality control** (QC) administration is "What is the inspection state ('in control', 'accepted', or 'rejected') of a batch submitted by company C 'last week' with part-id = 4711?"

Contrarily, OLAP databases ("analytical transaction processing") are designed to support the middle QC manager or the chief executives of a company with respect to planning, decisions and controlling. They were introduced in business from 1990 on [3]. They store business indicators as aggregated and grouped data by using statistical functions like *avg* (average), *sum, min, max*, and *counts* (frequency), combined with the grouping operator *group-by*. A typical indicator might be "Percentage of rejected items in incoming batches grouped by supplier-id, part-id, and quarter".

A major advantage of a database system is to hide details of the storage, the control of concurrent access and the maintenance of the data from the users. On the *physical level*, the database designer describes in a physical schema $S_p \subseteq S$ how the database is technically organized, stored and accessed. For example, in a distributed database system, data with lowest granularity is recorded on the shop floor. Daily grouping and summarizing of such measurements generates low-level aggregated data which is entered into a departmental data warehouse. Aggregation over departments produces an enterprise-wide view. Further details are skipped here, *cf.* [4]. The external schema $S_u \subseteq S$ provides a view for each authorized user u according to his individual ("personalized") profile. Views are important for enforcing data security and usage comfort. Access – as read or write – to the database is enabled by the (declarative) structured query language called *SQL* which is a *de facto* standard [4]. A conceptual schema $S_c \subseteq S$ is an abstract model of the real world focusing on the frame of discernment. It consists of $(\boldsymbol{R}, \boldsymbol{\Sigma})$ defining relations (tables) and related constraints. The DBMS furthermore supplies a data definition language to generate the tables and to provide a mapping from the conceptual level to the physical implementation.

We present now, an example of a conceptual schema of a QC database system for an adaptive acceptance sampling system which combines a skip-lot system and the Mil-Std. 105D [5]. This OLTP database system combines two principles of adaptability used in industry: lot-by-lot inspection with varying sample sizes or skipping lots according to the history of quality. We note that attributes which play the role of an identifier of objects (primary key) are underlined. The most elementary tables are those for suppliers, parts, and quality characteristics (attributes) which are an intrinsic part of any enterprise information system like SAP, Oracle Business Suite, and so on.

```
Supplier (supplier_id, name, info),
Part (part_id, name, description),
Attribute (attribute_id, name, description).
```

Furthermore a "supply list" is needed as a subset $\boldsymbol{L} \subseteq$ Supplier $\times$ Part:

```
Supply_Part (supplier_id, part_id, conditions).
```

The QC department is responsible for the design of the basis sampling plans of **attribute sampling** for reduced, normal and tightened inspection indexed by batch size, class and average quality level (AQL):

`Basis Sampling Plans (lower_batch_size, upper_batch_size, AQL_level, (sample_sizes, acceptance_numbers)).`

The same is true for switching rules which are an inherent part of adaptive systems. In the MISL (mixed MIL-Std 105D and Skip-Lot sampling system) system the states 0 to 2 correspond to the states in Mil-Std 105D, i.e., tightened, normal, and reduced inspection levels, while the inspection states 3 to 5 are skip-lot states with varying skip sizes or batch inspection frequencies:

`Switching_Rules (AQL_level, relaxation_number, skip_size3, skip_size4, skip_size5).`

As the AQL is assumed to be invariant with respect to suppliers, we have to know the AQL value for each quality characteristic and each part :

`Part_Attribute (part_id, attribute_id, AQL_level).`

Note that the test department, the test device, as well as the measurement method used in the inspection phase can easily be added as further attributes to that relation.

Any system, especially a dynamic one, needs information about the state of inspection of incoming batches. Assuming that incoming batches are strictly sequenced, we get:

`Inspection-state (supplier_id, part_id, attribute_id, batch_id, date, inspection level, inspection_no, skip_no).`

As the decision to inspect a batch or not and the **sample size** of batch inspection depends on the so-called history of quality of an attribute, i.e., on the sequence of former inspection states, we need to store the history of inspection for each attribute of every batch supplied by every supplier:

`History(date, batch_id, attribute_id, supplier_id, part_id, update_indicator, sample_size, acceptance_no, no_of_defects, batch_size).`

We summarize by representing the corresponding entity-relationship (ER) model in Figure 1.

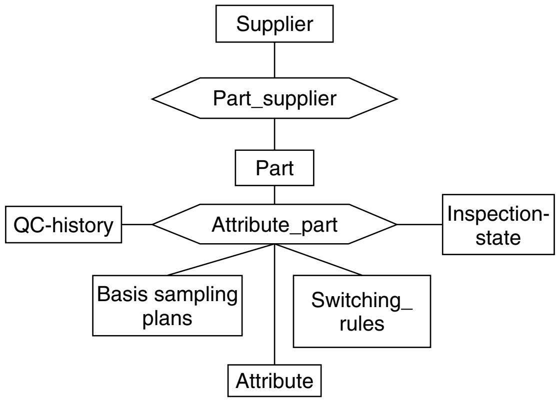

Figure 1 ER model for (adaptive) acceptance sampling

We illustrate the ease of querying of OLTP databases by checking whether or not batch b_1 of electronic switches supplied on October 27, 2007, passed inspection with respect to the tolerance of electric resistance (TER):

select no_of_defects ≤ acceptance_no
from History
where date = 2007-10-27 *and* batch_id = 'b 1' *and* attribute_id = 'TER'.

Data Warehousing for Acceptance Sampling

While querying of an operative database is user-friendly and is efficiently performed owing to tree-indexed structures and hashing [6], retrieval of aggregated data with range queries becomes more troublesome. Such deficiencies caused the development of OLAP databases called *data warehouses*. The main data structure is the data cube which is called *multiway table* in statistics [3]. It can be defined as a tuple (statistical function f, [summary attribute A], set of dimensions $\boldsymbol{D}$). The symbol [ ] means that this term is optional. Facts are composed of a statistical function and – optionally – a summary attribute like "number of defects" or "diameter". Dimensions are attributes which are not pair wise (strong or weak) functionally dependent [7] cf. Figure 2. For example, the data cube *Supplier-Report* (100 $*$ acceptance rate, ∅, {part_id, supplier_id, month/quarter/year}) represents the percentage of accepted incoming batches grouped by part, supplier, and time. Note that attributes like part or time may have domains which are partially ordered,

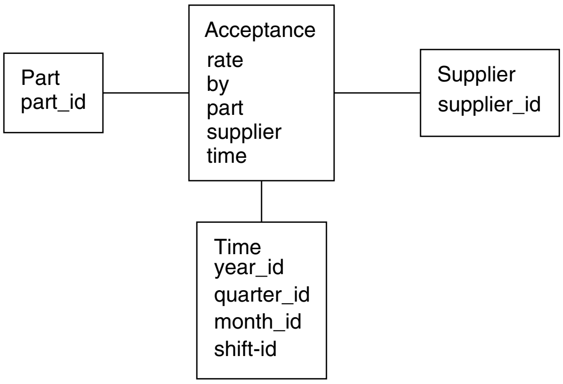

Figure 2 Data cube with one fact table and three-dimensional tables

i.e., form hierarchies. The cube operators needed are cube projection (dicing), cube selection (slicing) and hierarchy traversal (roll-up, drill-down). A cube, which can again be viewed as a table, can be materialized (permanently stored and updated) or virtual, i.e., computed "on the fly". The database administrator has to find the best trade-off between retrieval time and storage size.

Evidently, the mapping from OLTP to OLAP databases is not simple [1]. OLTP–OLAP data transformations and retrieval of data combine elements from the set of cube operators $\boldsymbol{O}$, the set of aggregation functions $\boldsymbol{F}$, and the set of transformations $\boldsymbol{T}$. Admissible operations should guarantee their statistical and semantic coherency. For instance, consider the transformation $\boldsymbol{T}$ of a consumption rate measured in l/100 km to miles/gallon, which is a nonlinear transformation. Testing cars with respect to consumption or mileage is usually done within and outside a city, and the measured rates of consumption or mileage are averaged. Of course, as a linear operator (average) and a nonlinear transformation $\boldsymbol{T}$ are not generally commutative, contradicting decision may be caused when either the consumption rate or the mileage rate is used [8].

Outlook

SAP Netweaver, IBM's database management system DB2, Microsoft's SQL server and Oracle's Business Suite offer a broad spectrum of automated **data collection** facilities and retrieval functionality for mass data. Although the border between querying and statistical data analysis is becoming vaguely more and more, there still remains a demand in industry for explorative data analysis, which can be handled by added-on **data mining** or "intelligent" data analysis tools [9, 10], for the underlying methodology.

References

[1] Lenz, H.-J. & Thalheim, B. (2005). OLAP schemata for correct applications, *Trends in Enterprise Architectures and Applications (TEAA)*, Springer, Heidelberg.

[2] Codd, E.F. (1970). A relational model for large shared databases, *Communications of The ACM* **13**(6), 377–387.

[3] Inmon, W.H. (1992). *Building the Data Warehouse*, Springer, New York.

[4] Elmasri, R. & Navathe, S.B. (1999). *Fundamentals of Database Systems*, 3rd Edition, Addision-Wesley.

[5] Lenz, H.-J. (1987). Design and implementation of a sampling inspection system for incoming batches based on relational databases, in *Frontiers in Statistical Quality Control*, H.-J. Lenz, G.B. Wetherill & P.-Th. Wilrich, eds, Physica, Heidelberg, Vol. 3, pp. 116–127.

[6] Gaede, V. & Günther, O. (1998). Multidimensional access methods, *ACM Computing Surveys* **30**(2), 170–231.

[7] Lehner, W., Albrecht, J. & Wedekind, H. (1998). Normal forms for multidimensional databases, *Proceedings of the 10th International Conference On Scientific and Statistical Data Management (SSDBM'98)*, IEEE, pp. 63–72.

[8] Hand, D. (1994). Deconstructing of statistical questions (with discussion), *Journal of the Royal Statistical Society. series A*, **157**, 317–356.

[9] Hand, D., Mannila, H. & Smyth, P. (2001). *Principles of Data Mining*, MIT Press, Cambridge.

[10] Berthold, M. & Hand, D. (eds) (1999). *Intelligent Data Analysis*, Springer, Berlin, Heidelberg.

HANS-J. LENZ