FlinkSQL

The SQL dialect for Flink

Khuong Duy Vu

Department of Informatics

ELTE University

This thesis is submitted for the degree of

Master Thesis

May 2015

Chapter 1

Introduction

Big Data

Data Streaming

CQL and related work

Features of stream processing languages(Fundamental of Stream Processing book page 110)

Flink and FlinkQL

Structure

[2014] Fundamentals of Stream Processing- Application Design, Systems, and Analytics.pdf [3]

Sliding Window Query Processing over Data Stream by Lukasz Golab

Chapter 2

Data Stream Model

Order

Ordering is rather important in Distributed system, specially Stream Processing System in particular.

In traditional model, there are a single program, one process, one memory space running on one CPU. Programs are written to be executed in an ordered fashion in queue: staring from the beginning, and then going toward the last. In distributed system, programs run in many machines across the network but try to reserve the order of the result. One of easiest way to define "correctness" is to say "it works like it would on a single machine". And that usually means that a) we run the same operations and b) that we run them in the same order - even if there are multiple machines. [Takada]

In theory, they are defined 2 types of orders: total order and partial order.

Definition 2.1 *Paritial order is a binary relation $\leq$ over a set P which is reflexive, anti-symmetric and transitive, i.e., which satisfies for all a, b and c in S (wiki):*

- *$a \leq a$ (reflexivity)*
- *if $a \leq b$ and $b \leq a$ then $a = b$ (antisymmetry)*
- *if $a \leq b$ and $b \leq c$ then $a \leq c$ (transitivity)*

Definition 2.2 *Total order is a binary relation "$\leq$" over a set S which is anti-symmetric and transitive and total. Therefore, total order is a partial order with totality*

- *$a \leq b$ or $b \leq a$ (totality)*

Total order “<” is strict on a set S if and only if $(S,<)$ has no non-comparable pairs:

$$\forall x,y \in S \Rightarrow x < y \cup y < x \tag{2.1}$$

In a totally ordered set, every two elements are comparable whereas in a partial ordered set, some pairs of elements are incomparable and hence we do not have the exact order of every item.

The natural state in a distributed system is partial order. Neither the network nor independent nodes make any guarantees about relative order; but at each node, you can observe a local order.

In streaming processing, one may discretize a stream into a discrete stream of windows. Therefore, we need to specify the order of elements to define which elements should be inserted into a given window. Furthermore, data stream involve the temporal order and the positional order [13] in stream.

The temporal order is induced by the timestamp of items in a stream. Using the value of timestamp, one can determine whether something happen chronologically before something else. Nevertheless, if some items happen simultaneously, they will have the same timestamp, then it may not be strict. For instance, there are two items with the same timestamp but system is required to take one only, the chosen is non-deterministic between two. Therefore, we may require a strict order.

The positional is a strict order induced by the position of item in stream. Two items may have the same timestamp but one may arrive before the other so that they have different positional orders. The positional order can be defined by arrival order or id of item regardless of explicit timestamp.

//TODO:The temporal order: Order. When I say that time is a source of order, what I mean is that:

we can attach timestamps to unordered events to order them we can use timestamps to enforce a specific ordering of operations or the delivery of messages (for example, by delaying an operation if it arrives out of order) we can use the value of a timestamp to determine whether something happened chronologically before something else.

Time

Time Domain The time domain $\mathbb{T}$ is a discrete, total ordered, countably infinite set of time instants $t \in \mathbb{T}$. We assume that $\mathbb{T}$ is bounded in the past , but not necessarily in the future. For the sake of simplicity, we will assume that the time domain is the domain of non-negative long integer ($\mathbb{T} = \mathbb{N}$) 0,1,2,3,..[8] and totally ordered.

2 type of times: system time t^{sys} (implicit) and application time t^{app} (explicit). These both take values from the time domain $\mathbb{T}$, but carry two different meaning.

Application Timestamps In many case, each item in stream contains an explicit source-assigned timestamp itself. In other words, the timestamp attribued will be a part of the stream schema. Consider a common log format for web application which contains a timestamp specifying when the action is taken place. A log line record user named *pablo* to get an image

```
216.58.209.174 user-identifier pablo [10/Oct/2000:13:55:36 -0700] \\
"GET /image.gif HTTP/1.0" 200 1234
```

Sequence number attribute could represent a timestamp instead.

System Timestamps If the item arrives at the system are not equipped with a timestamp, the system assigns a timestamp to each tuple, by default using the system's local clock. While this process generates a raw stream with system timestamps that can be processed like a regular raw stream with application timestamps, the user should be aware that application time and system time are not necessarily synchronized. [9]. As we mentions above, time domain is total ordered in local machine , but partial ordered across the system because of possible postpone on processing or asynchronous timestamp at different node. Moreover, since system timestamp is assigned implicitly by system , one may not notice it presence on schema.

Multiple items may have the same application timestamps but they will not arrive in the same order. Therefore, system will assign the different unique system timestamp based on their arrival. Sytem then can believe in the system timestamp as a strict total ordered basis for reasoning about arrival items to perform processing. For example, another log from different users arrive at system:

```
219.53.210.143 user-identifier fabio [10/Oct/2000:13:55:36 -0700] \\
"GET /image.gif HTTP/1.0" 200 1432
```

However, it might arrive after the first log for user *pablo* then system would response *pablo*'s first, instead of *fabio*'s request.

Note: more on Timestamp in Streams [6] page 13

Tuple

A tuple is a finite sequence of atomic values. Each tuple can be defined by a Schema corresponding to a composite type. Tuple can represend a relational tuple , a event or a record of sensor data and so on. [4]. For instance, the log format in previous example follow a schema

`<SourceIP, IdentityType, user, timestamp, action, response, packageSize >`

A data tuple is the fundamental, or atomic data item, embedded in a data stream and processed by an application. A tuple is similar to a database row in that it has a set of named and typed attributes. Each instance of an attribute is associated with a value.[3]. Furthermore, one can consider a tuple as a partial order mapping a finite subset of attribute names to atomic values[13]. A tuple consits of a set of (Atribute x Value) pairs such as $(SourceIP, 219.53.210.143)$

2.1 Stream Model

Based on time and tuple domain, basically, CQL [4] defines a data stream as

Definition 2.3 *A stream $\mathbb{S}$ is a countably continuous and infinite set of elements $< s,t > \in \mathbb{S}$, where s is a tuple belonging to the schema of $\mathbb{S}$ and $t \in \mathbb{T}$ is the timestamp of the element.*

There are several stream definitions based on the execution model of systems. On the previous definition, a timestamp attribute can be an non-strict total ordered application timestamp so that system may not rely on it to order the coming tuples to react on them. For this reason, stream can contain an extra physical identifier φ [12] such as increment tuple id to specify its order. The tuple with smaller id mean that they arrive and should be processed before the tuples with bigger id. Another way to identify the order of a tuple item is to separate the concept of application and system timestamp. In SECRET model [**?**], each stream item is composed of a tuple for event contents , an application timestamp, a system timestamp, and a batch-id value. The idea of batch-id is critical to SECRET system we do not mention in the thesis. In short, in practice, we assume that elements of a stream are totally ordered by the system timestamp and physical identifier.

In Apache Flink, for the simplicity, stream with timestamp function $f : \mathbb{TP} \rightarrow \mathbb{T}$ mapping a tuple to its application timestamp value.

Stream $s :< v >$ extends to $s :< v, t_{app}, t_{sys} >$, with $t_{app} = f(v)$, $t_{app} \in \mathbb{T}$, $t_{sys} \in \mathbb{T}$

//TODO Example:

Data Stream Properties

Data Stream may have the following properties [11]:

- They are considered as sequences of records, ordered by arrival time or by another ordered attributed such as generation time which is explicitly specified in schema, that arrive for processing over time instead of being available a priori. Totally order by time.

- They are emitted by a variety of external sources. Therefore, the system has no control over the arrival order or data rate, either within a stream or across multiple streams
- They are produced continually and, therefore, have unbounded, or at least unknown, length. Thus, a DSMS may not know if or when the stream "ends". We may set a time-out waiting for new event. Exceeding the time-out, the stream considerably ends.
- Typically, big volume of data arrives at very high speed so that data need to process on the fly. Once an element from a data stream model has been processed it is discarded or archived. Stream elements cannot be retrieved, unless it is explicitly stored in storage or memory, which typically is small relative to the size of the data stream. [6]

Stream Representations

We distinguish 2 kinds of streams: *base stream*(source stream) and *derived stream*. Base stream stream is produced by the sources whereas derived stream is produced by continuous queries and their operators[4]. For example, [11] page 17

In practice, base stream are almost always append-only , mean that previously arrived stream elements are never modified. However, derived stream may or may not be append-only . For example, [11]page 18

Logical stream ? [9]

Physical stream ? [9]

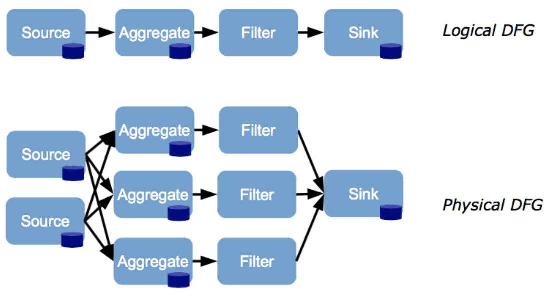


Figure 3: Extracting data parallelism from a logical DFG by replicating operators and segmenting their internal state in the corresponding physical DFG. The disk icon represents an operator's internal state.

Fig. 2.1 Stream Processing in Action. Henrique Andrade

Signals

[11] Data streams often describe underlying "signals". There are 4 ways in which a data stream can represent a signal

- aggregate model
- cash register model
- turnstile model
- reset model

2.2 Stream Window

Why do we need window?

It is often infeasible to maintain the entire history of the input data.

[3] a window over streaming data can be created, buffering individual tuples. Windows can be built according to a multitude of parameters that define what to buffer, resulting in many window variations. These variations differ in their policies with respect to evicting old data that should no longer be buffered, as well as in when to process the data that is already in the window.

Many stateful stream processing operators are designed to work on windows of tuples, making it a fundamental concept in stream processing. Therefore, a stream processing language must have rich windowing semantics to support the large diversity in how SPAs can consume data on a continuous basis.

From the system's point of view, it is infeasible to store an entire stream. From the user's point of view, recently arrived data may be more useful. This motivates the use of windows to restrict the scope of continuous queries. Windows may be classified according the following criteria. Window may be classified according the following criteria:

1. *Direction of movement*: fixed window, sliding window
2. *Definition of contents* time-based, count-based/tuple-based, partitioned windows, predicate window
3. *Frequency of movement* jumping window, mixed jumping window, tumbling window

(check Sliding Window Query Processing over Data Stream pages 25)

Windows are defined with respect to a timestamp or sequence number attribute representing a tuple's arrival time

Window

[8] A **Window** W over a stream $\mathbb{S}$ is a finite subset of $\mathbb{S}$ [8]

A **Time-based Window** $W = [o, c)$ over a stream $\mathbb{S}$ os a finite subset of $\mathbb{S}$ containing all data elements $s \in \mathbb{S}$ where $o \leq s.t^{sys} < c$ [8]

Window size and slide [8]

The notion of sliding windows requires at least an ordering on data stream elements. In many cases, the arrival orders of the elements suffices as an ïmplicit timestampättached to each data element. However, sometimes it is preferable to user ëxplicit timestampp̈rovided as part of data stream. Formally, we say that a data stream consists of a set of (tuple, timestamp) pairs. Timestamp attribute could be a traditional timestamp or it could be a sequence number - all that is required is that it come from a totally ordered domain with a distance metric. The ordering induced by the timestamp is used when selecting the data elements making up a sliding window.

Notes: in CQL tuple-based sliding window may be non-deterministic - and therefore may not be appropriate - when timestamp are not unique

Notes: Streambase: tuple-at-a-time

Notes: Tuple relational calculus by Edgar F. Codd

Notes: http://en.wikipedia.org/wiki/Relational_algebra

Window specification

Chapter 3

The execution semantic of Stream Processing in Flink

3.1 Heterogeneity

[8]

- Syntax
- Capability
- Execution Model

Order in stream

total order vs partial order

Chapter 4

Continuous Query Semantics and Operators

- Window alone come with Trigger
    - Window + Every:

    + Window : Eviction

    + Every: Trigger

    Eviction: number of events to keep (start of window)

    Trigger : when to start firing the function (end of window)

    startTime: end of the first Window

Chapter 5

Getting started

5.1 What is loren ipsum? Title with math σ

Definition 5.1 *This is a definition*

Definition 5.2 *This is a definition*

Lorem Ipsum is simply dummy text of the printing and typesetting industry (see Section 5.3). Lorem Ipsum [5] has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum [1, 7, 10].

The most famous equation in the world: $E^2 = (m_0c^2)^2 + (pc)^2$, which is known as the **energy-mass-momentum** relation as an in-line equation.

A *LaTeX class file* is a file, which holds style information for a particular LaTeX.

$$y_t = a_1y_{t-1} + a_2y_{t-2} + ... + a_py_{t-p} + \varepsilon_t \tag{5.1}$$

5.2 Why do we use loren ipsum?

It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less normal distribution of letters, as opposed to using 'Content here, content here', making it look like readable English. Many desktop publishing packages and web page editors now use

Lorem Ipsum as their default model text, and a search for ‘lorem ipsum’ will uncover many web sites still in their infancy. Various versions have evolved over the years, sometimes by accident, sometimes on purpose (injected humour and the like).

5.3 Where does it come from?

Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making it over 2000 years old. Richard McClintock, a Latin professor at Hampden-Sydney College in Virginia, looked up one of the more obscure Latin words, consectetur, from a Lorem Ipsum passage, and going through the cites of the word in classical literature, discovered the undoubtable source. Lorem Ipsum comes from sections 1.10.32 and 1.10.33 of "de Finibus Bonorum et Malorum" (The Extremes of Good and Evil) by Cicero, written in 45 BC. This book is a treatise on the theory of ethics, very popular during the Renaissance. The first line of Lorem Ipsum, "Lorem ipsum dolor sit amet..", comes from a line in section 1.10.32.

The standard chunk of Lorem Ipsum used since the 1500s is reproduced below for those interested. Sections 1.10.32 and 1.10.33 from “de Finibus Bonorum et Malorum" by Cicero are also reproduced in their exact original form, accompanied by English versions from the 1914 translation by H. Rackham

“Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum."

Section 1.10.32 of “de Finibus Bonorum et Malorum", written by Cicero in 45 BC: “Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?"

1914 translation by H. Rackham: “But I must explain to you how all this mistaken idea of denouncing pleasure and praising pain was born and I will give you a complete account of the system, and expound the actual teachings of the great explorer of the truth, the master-builder of human happiness. No one rejects, dislikes, or avoids pleasure itself, because it is pleasure, but because those who do not know how to pursue pleasure rationally encounter consequences that are extremely painful. Nor again is there anyone who loves or pursues or desires to obtain pain of itself, because it is pain, but because occasionally circumstances occur in which toil and pain can procure him some great pleasure. To take a trivial example, which of us ever undertakes laborious physical exercise, except to obtain some advantage from it? But who has any right to find fault with a man who chooses to enjoy a pleasure that has no annoying consequences, or one who avoids a pain that produces no resultant pleasure?"

Section 1.10.33 of “de Finibus Bonorum et Malorum", written by Cicero in 45 BC: “At vero eos et accusamus et iusto odio dignissimos ducimus qui blanditiis praesentium voluptatum deleniti atque corrupti quos dolores et quas molestias excepturi sint occaecati cupiditate non provident, similique sunt in culpa qui officia deserunt mollitia animi, id est laborum et dolorum fuga. Et harum quidem rerum facilis est et expedita distinctio. Nam libero tempore, cum soluta nobis est eligendi optio cumque nihil impedit quo minus id quod maxime placeat facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum hic tenetur a sapiente delectus, ut aut reiciendis voluptatibus maiores alias consequatur aut perferendis doloribus asperiores repellat."

1914 translation by H. Rackham: “On the other hand, we denounce with righteous indignation and dislike men who are so beguiled and demoralized by the charms of pleasure of the moment, so blinded by desire, that they cannot foresee the pain and trouble that are bound to ensue; and equal blame belongs to those who fail in their duty through weakness of will, which is the same as saying through shrinking from toil and pain. These cases are perfectly simple and easy to distinguish. In a free hour, when our power of choice is untrammelled and when nothing prevents our being able to do what we like best, every pleasure is to be welcomed and every pain avoided. But in certain circumstances and owing to the claims of duty or the obligations of business it will frequently occur that pleasures have to be repudiated and annoyances accepted. The wise man therefore always holds in these matters to this principle of selection: he rejects pleasures to secure other greater pleasures, or else he endures pains to avoid worse pains."

Chapter 6

My second chapter

6.1 Reasonably long section title

6.1.1 This is a short title

I’m going to randomly include a picture Figure 6.1.

If you have trouble viewing this document contact Krishna at: kks32@cam.ac.uk or raise an issue at https://github.com/kks32/phd-thesis-template/

Enumeration

1. The first topic is dull
2. The second topic is duller
   (a) The first subtopic is silly
   (b) The second subtopic is stupid
3. The third topic is the dullest

itemize

- The first topic is dull
- The second topic is duller
  - The first subtopic is silly

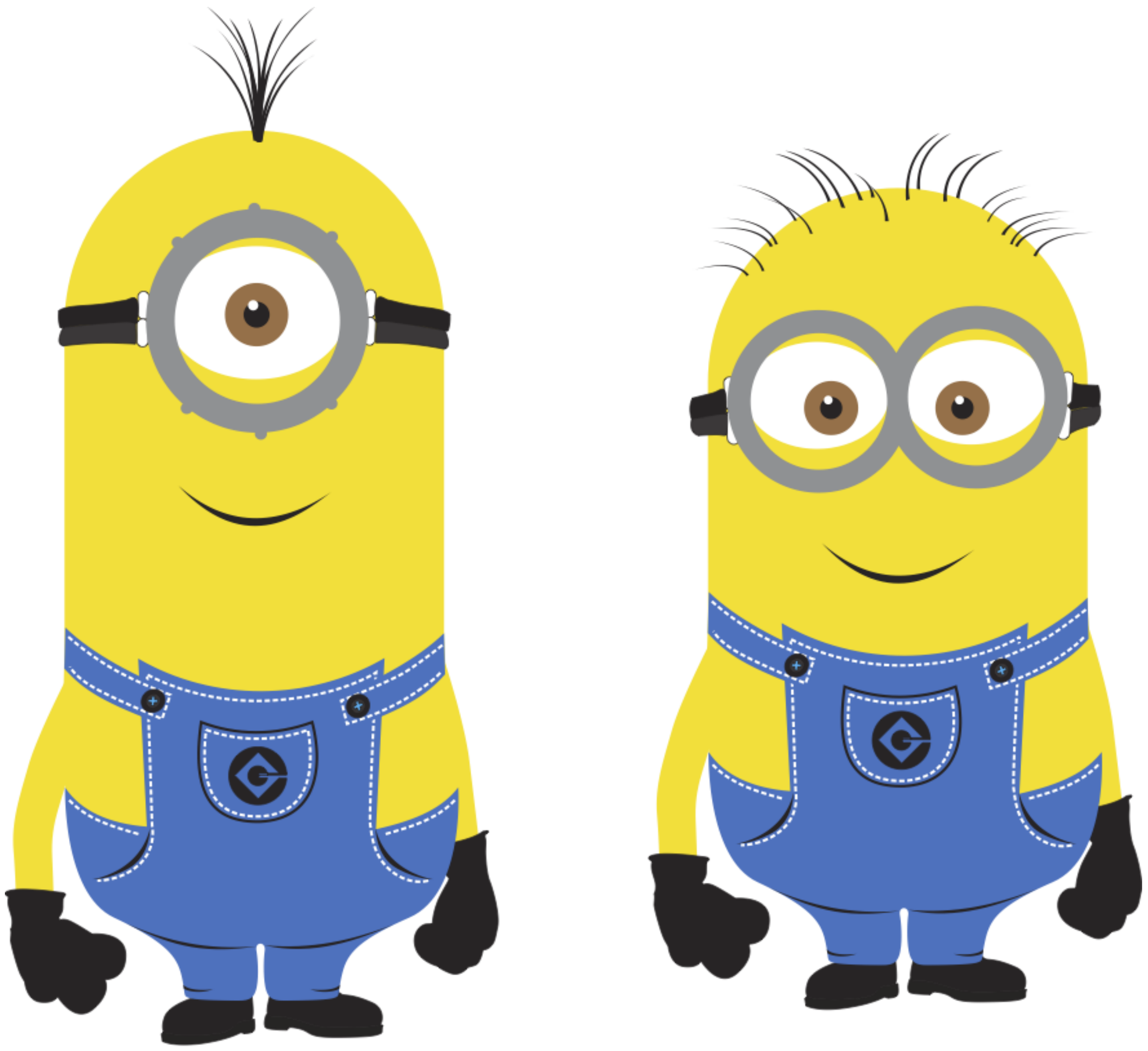


Fig. 6.1 This is just a long figure caption for the minion in Despicable Me from Pixar

  - The second subtopic is stupid

- The third topic is the dullest

description

The first topic is dull

The second topic is duller

  The first subtopic is silly

  The second subtopic is stupid

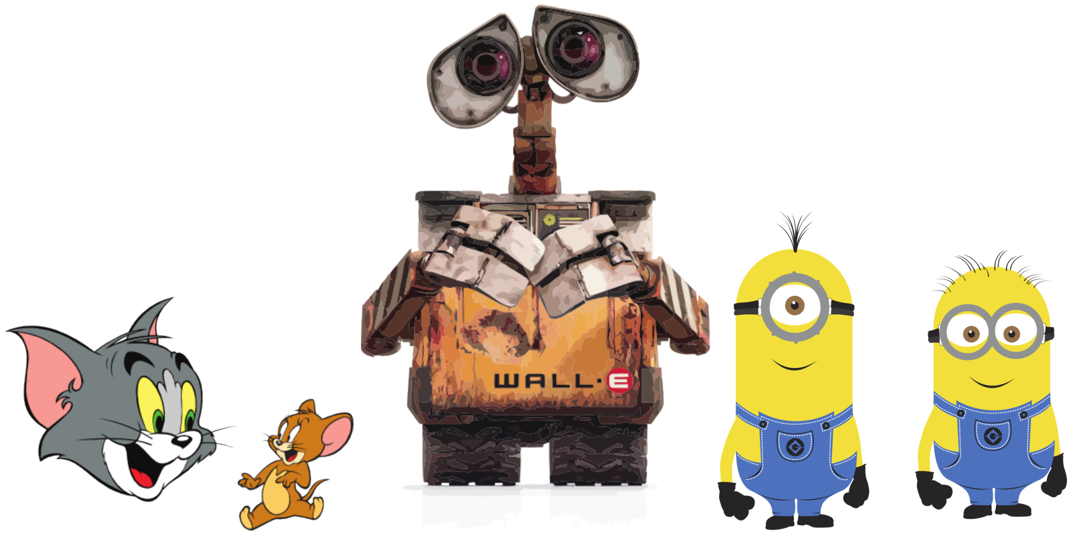
The third topic is the dullest

6.2 Hidden section

Lorem ipsum dolor sit amet, *consectetur adipiscing elit.* In magna nisi, aliquam id blandit id, congue ac est. Fusce porta consequat leo. Proin feugiat at felis vel consectetur. Ut tempus ipsum sit amet congue posuere. Nulla varius rutrum quam. Donec sed purus luctus, faucibus velit id, ultrices sapien. Cras diam purus, tincidunt eget tristique ut, egestas quis nulla. Curabitur vel iaculis lectus. Nunc nulla urna, ultrices et eleifend in, accumsan ut erat. In ut ante leo. Aenean a lacinia nisl, sit amet ullamcorper dolor. Maecenas blandit, tortor ut scelerisque congue, velit diam volutpat metus, sed vestibulum eros justo ut nulla. Etiam nec ipsum non enim luctus porta in in massa. Cras arcu urna, malesuada ut tellus ut, pellentesque mollis risus.Morbi vel tortor imperdiet arcu auctor mattis sit amet eu nisi. Nulla gravida urna vel nisl egestas varius. Aliquam posuere ante quis malesuada dignissim. Mauris ultrices tristique eros, a dignissim nisl iaculis nec. Praesent dapibus tincidunt mauris nec tempor. Curabitur et consequat nisi. Quisque viverra egestas risus, ut sodales enim blandit at. Mauris quis odio nulla. Cras euismod turpis magna, in facilisis diam congue non. Mauris faucibus nisl a orci dictum, et tempus mi cursus.

Etiam elementum tristique lacus, sit amet eleifend nibh eleifend sed [1]. Maecenas dapibu augue ut urna malesuada, non tempor nibh mollis. Donec sed sem sollicitudin, convallis velit aliquam, tincidunt diam. In eu venenatis lorem. Aliquam non augue porttitor tellus faucibus porta et nec ante. Proin sodales, libero vitae commodo sodales, dolor nisi cursus magna, non tincidunt ipsum nibh eget purus. Nam rutrum tincidunt arcu, tincidunt vulputate mi sagittis id. Proin et nisi nec orci tincidunt auctor et porta elit. Praesent eu dolor ac magna cursus euismod. Integer non dictum nunc. 6.2c

[1]My footnote goes blah blah blah! …



(a) Tom and Jerry (b) Wall-E (c) Minions

Fig. 6.2 Best Animations

Subplots

I can cite Wall-E (see Fig. 6.2b) and Minions in despicable me (Fig. 6.2c) or I can cite the whole figure as Fig. 6.2

Chapter 7

My third chapter

7.1 First section of the third chapter

And now I begin my third chapter here . . .

And now to cite some more people Ancey et al. [2], Read [14]

7.1.1 First subsection in the first section

. . . and some more

7.1.2 Second subsection in the first section

. . . and some more . . .

First subsub section in the second subsection

. . . and some more in the first subsub section otherwise it all looks the same doesn't it? well we can add some text to it . . .

7.1.3 Third subsection in the first section

. . . and some more . . .

First subsub section in the third subsection

. . . and some more in the first subsub section otherwise it all looks the same doesn't it? well we can add some text to it and some more and some more and some more and some more and some more and some more and some more . . .

Second subsub section in the third subsection

. . . and some more in the first subsub section otherwise it all looks the same doesn't it? well we can add some text to it . . .

7.2 Second section of the third chapter

and here I write more . . .

7.3 The layout of formal tables

This section has been modified from "Publication quality tables in LaTeX*" by Simon Fear.

The layout of a table has been established over centuries of experience and should only be altered in extraordinary circumstances.

When formatting a table, remember two simple guidelines at all times:

1. Never, ever use vertical rules (lines).
2. Never use double rules.

These guidelines may seem extreme but I have never found a good argument in favour of breaking them. For example, if you feel that the information in the left half of a table is so different from that on the right that it needs to be separated by a vertical line, then you should use two tables instead. Not everyone follows the second guideline:

There are three further guidelines worth mentioning here as they are generally not known outside the circle of professional typesetters and subeditors:

3. Put the units in the column heading (not in the body of the table).
4. Always precede a decimal point by a digit; thus 0.1 *not* just .1.
5. Do not use 'ditto' signs or any other such convention to repeat a previous value. In many circumstances a blank will serve just as well. If it won't, then repeat the value.

A frequently seen mistake is to use '\begin{center}' . . . '\end{center}' inside a figure or table environment. This center environment can cause additional vertical space. If you want to avoid that just use '\centering'

Table 7.1 A badly formatted table

	Species I		Species II	
Dental measurement	mean	SD	mean	SD
I1MD	6.23	0.91	5.2	0.7
I1LL	7.48	0.56	8.7	0.71
I2MD	3.99	0.63	4.22	0.54
I2LL	6.81	0.02	6.66	0.01
CMD	13.47	0.09	10.55	0.05
CBL	11.88	0.05	13.11	0.04

Table 7.2 A nice looking table

Dental measurement	Species I		Species II	
	mean	SD	mean	SD
I1MD	6.23	0.91	5.2	0.7
I1LL	7.48	0.56	8.7	0.71
I2MD	3.99	0.63	4.22	0.54
I2LL	6.81	0.02	6.66	0.01
CMD	13.47	0.09	10.55	0.05
CBL	11.88	0.05	13.11	0.04

Table 7.3 Even better looking table using booktabs

Dental measurement	Species I		Species II	
	mean	SD	mean	SD
I1MD	6.23	0.91	5.2	0.7
I1LL	7.48	0.56	8.7	0.71
I2MD	3.99	0.63	4.22	0.54
I2LL	6.81	0.02	6.66	0.01
CMD	13.47	0.09	10.55	0.05
CBL	11.88	0.05	13.11	0.04

References

[1] Abramovich, Y. A., Aliprantis, C. D., and Burkinshaw, O. (1995). Another characterization of the invariant subspace problem. *Operator Theory in Function Spaces and Banach Lattices.* The A.C. Zaanen Anniversary Volume*, Operator Theory: Advances and Applications*, 75:15–31. Birkhäuser Verlag.

[2] Ancey, C., Coussot, P., and Evesque, P. (1996). Examination of the possibility of a fluid-mechanics treatment of dense granular flows. *Mechanics of Cohesive-frictional Materials*, 1(4):385–403.

[3] Andrade, H. C. M., Gedik, B., and Turaga, D. S. (2014). *Fundamentals of Stream Processing*. Cambridge University Press. Cambridge Books Online.

[4] Arasu, A., Babu, S., and Widom, J. (2006). The cql continuous query language: Semantic foundations and query execution. *The VLDB Journal*, 15(2):121–142.

[5] Aupetit, B. (1991). *A Primer on Spectral Theory*. Springer-Verlag, New York.

[6] Babcock, B., Babu, S., Datar, M., Motwani, R., and Widom, J. (2002). Models and issues in data stream systems. In *Proceedings of the Twenty-first ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '02, pages 1–16, New York, NY, USA. ACM.

[7] Conway, J. B. (1990). *A Course in Functional Analysis*. Springer-Verlag, New York, second edition.

[8] Dindar, N., Tatbul, N., Miller, R. J., Haas, L. M., and Botan, I. (2013). Modeling the execution semantics of stream processing engines with secret. *The VLDB Journal*, 22(4):421–446.

[9] Krämer, J. and Seeger, B. (2009). Semantics and implementation of continuous sliding window queries over data streams. *ACM Trans. Database Syst.*, 34(1):4:1–4:49.

[10] Ljubič, J. I. and Macaev, V. I. (1965). On operators with a separable spectrum. *Amer. Math. Soc. Transl. (2)*, 47:89–129.

[11] Lukasz Golab, M. T. O. (2010). Data stream management.

[12] Petit, L., Labbé, C., and Roncancio, C. L. (2010). An algebric window model for data stream management. In *Proceedings of the Ninth ACM International Workshop on Data Engineering for Wireless and Mobile Access*, MobiDE '10, pages 17–24, New York, NY, USA. ACM.

[13] Petit, L., Labbé, C., and Roncancio, C. L. (2012). Revisiting formal ordering in data stream querying. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, SAC '12, pages 813–818, New York, NY, USA. ACM.

[14] Read, C. J. (1985). A solution to the invariant subspace problem on the space l_1. *Bull. London Math. Soc.*, 17:305–317.

[Takada] Takada, M. *Distributed Systems For Fun and Profit*.

Appendix A

Notices

NOTE file:///Volumes/Data/WORKSPACE/git/thesis/rrd-antlr4/target/output/Json/index.html

Installation

1. Mount the ISO file in the mnt directory

   ```
   mount -t iso9660 -o ro,loop,noauto /your/texlive####.iso /mnt
   ```

2. Install wget on your OS (use rpm, apt-get or yum install)

3. Run the installer script install-tl.

   ```
   cd /your/download/directory
   ./install-tl
   ```

4. Enter command 'i' for installation

5. Post-Installation configuration:
   http://www.tug.org/texlive/doc/texlive-en/texlive-en.html#x1-320003.4.1

6. Set the path for the directory of TexLive binaries in your .bashrc file

For 32bit OS

For Bourne-compatible shells such as bash, and using Intel x86 GNU/Linux and a default directory setup as an example, the file to edit might be

```
edit $~/.bashrc file and add following lines
PATH=/usr/local/texlive/2011/bin/i386-linux:$PATH;
export PATH
MANPATH=/usr/local/texlive/2011/texmf/doc/man:$MANPATH;
export MANPATH
INFOPATH=/usr/local/texlive/2011/texmf/doc/info:$INFOPATH;
export INFOPATH
```

For 64bit OS

```
edit $~/.bashrc file and add following lines
PATH=/usr/local/texlive/2011/bin/x86_64-linux:$PATH;
export PATH
MANPATH=/usr/local/texlive/2011/texmf/doc/man:$MANPATH;
export MANPATH
INFOPATH=/usr/local/texlive/2011/texmf/doc/info:$INFOPATH;
export INFOPATH
```

Appendix B

Installing the CUED class file

LaTeX.cls files can be accessed system-wide when they are placed in the <texmf>/tex/latex directory, where <texmf> is the root directory of the user's TeXinstallation. On systems that have a local texmf tree (<texmflocal>), which may be named "texmf-local" or "localtexmf", it may be advisable to install packages in <texmflocal>, rather than <texmf> as the contents of the former, unlike that of the latter, are preserved after the LaTeXsystem is reinstalled and/or upgraded.

It is recommended that the user create a subdirectory <texmf>/tex/latex/CUED for all CUED related LaTeXclass and package files. On some LaTeXsystems, the directory look-up tables will need to be refreshed after making additions or deletions to the system files. For TeXLive systems this is accomplished via executing "texhash" as root. MIKTeXusers can run "initexmf -u" to accomplish the same thing.

Users not willing or able to install the files system-wide can install them in their personal directories, but will then have to provide the path (full or relative) in addition to the filename when referring to them in LaTeX.