	We want to make a certain kind of datasets in mathematics more useful. I'll pick up where Michael left off on Wednesday. I will first explain which kind of datasets need the most help and what I mean by useful.
	NEXT SLIDE!
	If I counted right, this is the third time you are seeing the Tetrapod. On Wednesday, you heard that research data is the next big thing. Unfortunately, it's the other three endpoints of the Tetrapod that are pretty well taken care of.
	NEXT SLIDE!
	Let's zoom into tabulation. This diagram is an updated version of the one you've seen on Wednesday. We've heard a lot about **symbolic data** this week. All the formal mathematics and computation fits in here. We've also seen some **linked data**, which relate to data in library science and ontologies. The missing spoke from Wednesday is **narration** and the last one is **concrete data**. Even here, three of the four parts are reasonably well take care of. The situation for concrete data however, could at best be called interesting — I'll show you why after 5 slides.
	NEXT SLIDE!
	Concrete stands for "not abstract", as in **the** interesting cubic graph on 10 vertices, as opposed to **an** abstract graph with a vertex and edge set. Michael mentioned the tentative characterisation for concrete mathematical data with representation theorems which specify how the objects can be uniquely represented in a database. He mentioned an example with elliptic curves. Here's an example with graphs. You can label the vertices of a graph of order n with n numbers and then represent the graph as a list of edges or pairs of numbers. You can then further encode that as a string. There are some popular encodings of this sort. If you used a canonical labeling, then the representation is nice. In particular, graphs with the same encoding are necessarily also isomorphic. And anyone who has heard of the graph isomorphism problem knows I'm sweeping some details under the rug. Apart from explaining what concrete data are, I'm also trying to make (or repeat) a point here. Mathematical meaning can be quite far removed from the encoding used in the database.
	NEXT SLIDE!

	The datasets look a little bit like this. You have a simple list of objects (in this case, edge-transitive graphs) such that each object comes equipped with information about a non-negative number of it's properties (in this case, graph-theoretic and symmetry related).
	NEXT SLIDE!
	Now, let's return to my one sentence summary and expand on it. Concrete data are more common in mathematics than one would expect. By useful, I primarily mean the things that mathematicians care about: make the datasets easier to share and searchable, ideally through a math-level interface. To this we tack on the buzzword compliant (but relevant!) features of findable, accessible, interoperable, and reusable. I'll now give you a rough outline of the landscape. Then I'll tell you about our vision for the framework.
	NEXT SLIDE!
	Mathematicians actually have a long history of producing data. Moreover, they want to share it. They store the things that are interesting and/or hard to compute. They then use the data as a reference and to better understand the structure of objects they study, such as a source of examples. Here are two early examples. You probably all know about the logarithm tables. For the early 20th century example, we have the Foster census. In 1930, Ronald Foster started collecting a class of symmetric cubic graphs and produced a list of such graphs up to order 512 (207), only missing a handful.
	NEXT SLIDE!
	In the last few decades, computation, digital storage and online collaboration became easier, and we have seen the rise of a few prominent mathematical databases and a multitude of smaller datasets. Probably the most famous example is The Online Encyclopedia of Integer Sequences. Its core is a list of integer sequences. In addition to the sequences, the encyclopedia contains bibliographic references, formulas, code for computer algebra systems, references to related sequences, and more.
	NEXT SLIDE!

	But researchers are busy people and often, the situation is more like this. On the left we have an html table generated in Magma. On the other side, we have links to Magma code files. In both cases, we are dealing with a fragile representation of data that makes it hard to reuse. Moreover, the sustainability of a project depends on one person (for some value of one). Projects like these are our main target. I'm making two assumptions. First, there are enough mathematicians would like to have better tools to work with data to justify building a framework. I just gave a 10 minute version of this talk to a room full of combinatorialists last week, and they seemed to agree. Second, most datasets have simple structure and can, with the right tools, be well taken care of with a fraction of the work that goes into the larger projects.
	NEXT SLIDE!
	From what I can tell, mathematicians have the following main concerns: • They need a place to **host** their data. They are generally aware that their work website is not the best place for this, but for many, it turns out to be the most feasible thing to do. It needs to be **easy way to submit** a dataset to the system — the work needed needs to have a corresponding payoff in things like dataset usability and visibility. • They would like to **look things up, filter subsets of the data, and run further computations**. Especially to run computations, they would ideally like to access the data from their favourite software (such as computer algebra system).
	NEXT SLIDE!
	We believe the way to go is a hosting platform for math data. To the best of our knowledge, the necessary core components are highly reusable. There are other benefits to doing it like this: standardized interfaces and cross-database sharing and linking, enabling math-level interfaces. In particular, it should be easier to make math data FAIR.
	NEXT SLIDE!

	The pre-requirement is that everything needs to be well **described** and **documented**. To illustrate one reason why, let me show you the second slide again. Even experts would appreciate an explanation with this table here, as the mathematical meaning of the fields is often unclear to all but a few experts. We need to provide at least the simplest web-based search interface. We also need to do all that with an eye towards mathematics and mathematics aware services. Let me illustrate. It is all fine and dandy while the users want to get all graphs with 5 vertices or all bipartite graphs. But some queries that mathematicians really want to ask are closer to “all arc-transitive graphs of order that is a product of two primes”?
	NEXT SLIDE!
	Here’s an outline of our vision of MathDataHub. We have a database-interface generator that takes a dataset description and produces all the necessary dataset specific infrastructure. In the same ecosystem, we also have collaborative libraries of codecs (mappings between formats) and presenters (for displaying matrices as matrices in the browser).
	NEXT SLIDE!
	We provide the vital ingredient: a high-level data description framework MDDL. Given a dataset description in MDDL, we can generate a full website stack. This simple website setup is already a step forward compared to how most datasets are currently shared. Since we aren’t using any of the mathematics from the schema theory yet, we could be using a generic framework. However, we do intend to go beyond generic interfaces and for that, we need a stronger description. The work so far is relevant for two reasons. 1. Mathematicians will be pretty happy if we just get the current system up and running. 2. We are confident that the current system is a prerequisite for higher flying goals. If I still have some time left, I can show a few details.
	NEXT SLIDE!
	We now have a reasonable interface generator for simple datasets. The datasets we can cover already include a large number of the datasets out there. We also provide the basic codecs to seed the codec library. To no-one surprise, the dataset descriptions are written in MMT. For the web interface we build on an existing project called DiscreteZOO.

	In our setup, the schema theories (below) are the main theories of interest. The declarations in schema theories correspond to the columns/properties. If one dataset builds on another the base dataset gets included in the schema theory of the one that is building on it. The meta theory MDDL is a collection of theories that define the mathematical types, the database types, and the codecs translating between them. It is important to understand that MathData (the theory covering mathematical types) is not a symbolic library of formalized or implemented mathematics; it is much simpler. Foundational questions are almost entirely irrelevant for mathematical database schemas. We also do not define any of the types or operations. Formal definitions are irrelevant because they would never be applied anyway: all computation happens in the database.
	NEXT SLIDE!
	Suppose you had a dataset with matrices and you computed whether they were orthogonal. This is roughly how your schema theory might look like. It might appear as an overly complicated way of describing a dataset and indeed one of the goals for the future is to reduce the boilerplate. Essentially, a schema theory is a list of declarations c colon A, where c is a name and A is the mathematical type of c. Every declaration also needs information (as metadatum) about the corresponding database type and the codec.
	NEXT SLIDE!
	This is how the website looks like. You have the available filters in the box on the right. The filters you select go into the left box. Over here, you get immediate feedback on how many matches your search has in the database. Finally, the search results are displayed in a simple table.