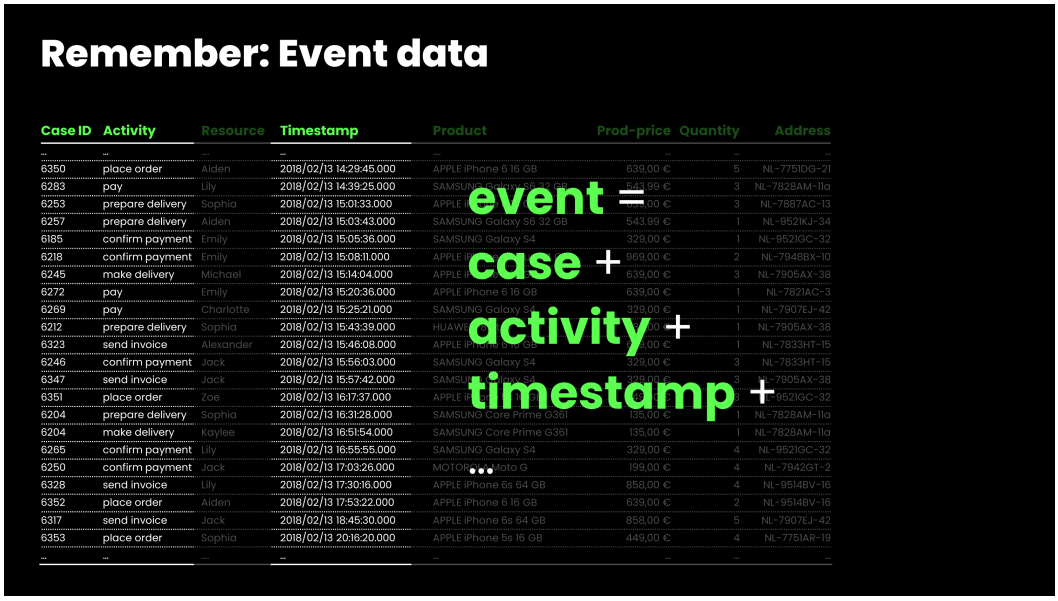
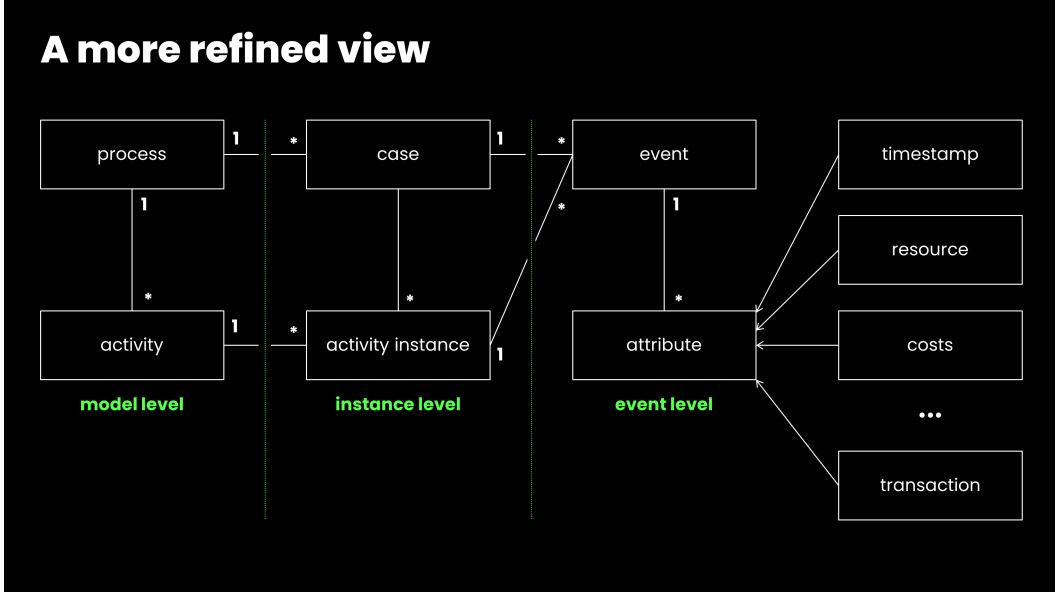
**проверка доступа истекает 18 февр. 2024 г.**
Вы теряете весь доступ к этому курсу, в том числе и ваш прогресс в учебе 18 февр. 2024 г..

Event data need to be *extracted* from information systems used to support the processes that need to be analyzed. Customer Relationship Management (CRM), Enterprise Resource Planning (ERP), and Supply Chain Management (SCM) systems store events. Consider, for example, the *Purchase-to-Pay* (P2P) and *Order-to-Cash* (O2C) processes that most organizations have. The P2P process is there to input products and services from suppliers into the organization. The O2C process is there to output products and services to customers (including invoicing and handling payments). These processes are supported by systems like SAP S/4HANA, Oracle E-Business Suite, and Microsoft Dynamics 365. These systems have many database tables storing information about these and other operational processes. This means that events are *scattered over many different tables* and often also over *different systems*. Also, when systems are domain-specific (e.g., healthcare) or organization-specific (i.e., homebrew software), there are typically many database tables containing events. Note that most database tables managed by contemporary information systems contain dates or timestamps. Some systems also store changes in these tables (e.g., the so-called "change tables" in SAP).

Hence, there are plenty of events. However, the challenge is to extract and convert these into a format that can be used for process mining. This is an integral part of any process mining effort and typically be time-consuming. In the simplest setting, events are represented by a *case identifier*, an *activity name*, and a *timestamp*. There may be optional attributes like resource, location, cost, etc.




For process discovery and conformance checking, we focussed on control flow, i.e., the ordering of activities. However, in reality, there are much more data.




Events may have any number of attributes. However, in the classical setting, events have at least a timestamp, refer to an activity, and belong to a case. Sometimes we use the notion of an *activity instance*, i.e., the execution of an activity for a specific case. One activity instance may consist of several events, e.g., the start and completion of the activity instance. Therefore, an event may have a transaction attribute. Also, cases may have attributes (not shown) and belong to a process. Traditionally, we assume that case attributes do not change over time (e.g., the birth date of a patient) and that a case belongs to precisely one process.

XES (eXtensible Event Stream) is the IEEE standard for storing traditional event data, as described above. The IEEE Task Force on Process Mining started to work on this in September 2010, and it became an official IEEE standard in 2016. XES supports the standard notions, including activity instances. Classifiers are used to attach labels to events. There is always at least one classifier, and by default, this is the activity name. However, it is also possible to project events onto resources, locations, departments, etc., or combinations of attributes. To handle activities that take time, XES provides the possibility to represent lifecycle information and connect events through activity instances. As mentioned, an activity instance is a collection of related events that together represent the execution of an activity for a case. The XES lifecycle model distinguishes between the following types of events: schedule, assign, withdraw, reassign, start, suspend, resume, abort, complete, autoskip, and manualskip. If event data contain such lifecycle information, it is possible to measure service times, waiting times, synchronization times, idle times, etc.

Although the corresponding traditional event logs (stored using XES or not) provide a clear starting point for process mining, they also impose a few limitations. When extracting the event log, one is forced to pick a *viewpoint* in the form of a *case identifier*. This means data extraction needs to be done for every view. Also, events cannot refer to multiple objects. This complicates the extraction of event data from source systems.