**Czym jest UML - podstawowe diagramy w UML'u.**

**UML** – graficzny system wizualizacji, specyfikowania oraz dokumentowania składników systemów informatycznych. To notacja umożliwiająca zaprezentowanie systemu w sposób graficzny, za pomocą diagramów. Modele zapisane w języku UML prezentują system od ogółu do szczegółu, umożliwiając oglądanie modelu systemu z wybraną w danym momencie szczegółowością.

**Diagramy:**

**Diagram przypadków użycia** – przedstawia funkcjonalność systemu wraz z jego otoczeniem. Pozwalają na graficzne zaprezentowanie własności systemu tak, jak są one widziane po stronie użytkownika.

**Diagram pakietów** – strukturalny diagram prezentujący pakiety i relacje zachodzące pomiędzy nimi. Pozwala na modelowanie systemu na wysokim poziomie abstrakcji, gdyż pakiety reprezentują ogromną liczbę klas, interfejsów, diagramów – pozwala to wyeksponować tylko zasadnicze funkcje systemu.

**Diagram klas** – obrazuje pewien zbiór klas, interfejsów i kooperacji oraz związki między nimi. Jest on grafem złożonym z wierzchołków (klas, interfejsów, kooperacji) i łuków (reprezentowanych przez relacje). Uwypukla związki między klasami, pomijając pozostałe charakterystyki.

**Diagram aktywności** – diagram interakcji służący do modelowania dynamicznych aspektów systemu. Jego zasadniczą funkcją jest przedstawienie sekwencji kroków, które są wykonywane przez modelowany fragment systemu. Diagram czynności jest stosowany do modelowania behawioralnych aspektów systemu i obrazuje strumień kolejno wykonywanych czynności.

**Diagram maszyny stanowej** – przedstawia maszynę stanową, która zawiera proste stany i przejścia pomiędzy nimi.

**Diagram sekwencji** – służy do prezentowania interakcji pomiędzy obiektami wraz z uwzględnieniem w czasie komunikatów, jakie są przesyłane pomiędzy nimi. Zasadniczym zastosowaniem diagramów sekwencji jest modelowanie zachowania systemu w kontekście scenariuszy przypadków użycia. Diagramy sekwencji pozwalają uzyskać odpowiedź na pytanie, jak w czasie przebiega komunikacja pomiędzy obiektami.

**Diagram komponentów** – służy do ilustracji organizacji i zależności pomiędzy komponentami. Prezentuje system na wyższym poziomie abstrakcji niż diagram klas, gdyż każdy z komponentów może być implementacją jednej lub większej liczby klas.

**Projektowanie baz danych - diagramy ERD, jak z tych diagramów tworzymy schemat relacyjnej bazy danych (czyli jak encje i związki zamieniamy w tabele bazodanowe - jakie są reguły)**

**Encja –** istniejący obiekt rozróżnialny od innych bytów tego samego typu (np. student, wydział, katedra).

**ERD** - Entity Relationship Diagrams – diagramy związków encji, graficzna prezentacja związków pomiędzy zbiorami encji.

**Atrybut** – informacja charakteryzująca encję (np. wiek, wydział).

**Rekord** – zestaw danych opisywanych przez atrybut (np. wiek = 25 i wydział = Informatyka).

**Tabela** – wydzielony logicznie zbiór danych, również reprezentacja encji.

**Związek** - nazwane powiązanie pomiędzy dwoma encjami (np. student > wydział). Każdy związek składa się z nazwy, liczebności (krotności) oraz opcjonalności.

**Diagramy ERD** wykorzystywane są do graficznej prezentacji bazy danych. Dobrze przygotowany diagram ERD pozwala na zrozumienie struktury danych, przygotowania późniejszej strategii optymalizacji bazy oraz stanowi podstawową dokumentację systemu przechowywania informacji.

**Relacje pomiędzy tabelami:**

Rozróżnia się trzy relacje jakie mogą zachodzić pomiędzy dwoma encjami.

**Relacja jeden do jednej** – jeden rekord w pierwszej tabeli odpowiada dokładnie jednemu rekordowi w drugiej tabeli.

**Relacja jeden do wielu** – jeden rekord w pierwszej tabeli odpowiada wielu rekordom w drugiej tabeli. (relacja najczęściej stosowana)

**Relacja wiele do wielu** – kilka rekordów z pierwszej tabeli odpowiada wielu rekordom w drugiej tabeli.

**Reguły transformacji encji:**

1. Encja jest odzwierciedlana w relację. Nazwa encji jest odwzorowywana w nazwę relacji.
2. Atrybut encji jest odzwierciedlany w atrybut relacji. Nazwy atrybutów encji są odwzorowywane w nazwy atrybutów relacji.
3. Typ danych atrybutu encji jest odwzorowywany w odpowiadający mu typ danych atrybutu relacji.
4. Unikalny identyfikator encji jest transformowany w klucz podstawowy relacji
5. Obowiązkowość atrybutów encji jest reprezentowana w relacji w postaci ograniczenia NOT NULL zdefiniowanego na atrybucie relacji odpowiadającym atrybutowi encji.
6. Opcjonalność atrybutów encji jest reprezentowana w relacji w postaci ograniczenia NULL zdefiniowanego na atrybucie relacji odpowiadającym atrybutowi encji.
7. Oganiczenia integralnościowe dla atrybutów encji (unikalność, zawężenie dziedziny) są transformowane do odpowiadających im ograniczeń integralnościowych relacji.

**Reguły transformacji związków:**

Związek binarny 1:1 jednostronnie obowiązkowy transformuje się do klucza obcego w tabeli po stronie związku obowiązkowego

Związek binarny 1:1 obustronnie opcjonalny transformuje się do klucza obcego w tabeli o mniejszym rozmiarze.

**Reguły transformacji związku 1:M są następujące:**

1. Klucz obcy jest dodawany do relacji po stronie "wiele" niezależnie od opcjonalności, czy obowiązkowości tego związku.

2. Ograniczenia referencyjne (tj. definiujące klucz obcy) są definiowane dla atrybutu reprezentującego klucz obcy.

3. Obowiązkowość związku po stronie "wiele" jest reprezentowana przez ograniczenie NOT NULL definiowane na kluczu obcym relacji.

4. Opcjonalność związku po stronie "wiele" jest reprezentowana przez ograniczenie NULL definiowaną na kluczu obcym relacji.

5. Opcjonalność lub obowiązkowość związku po stronie "jeden" nie jest odwzorowywana w modelu relacyjnym

**Reguły transformacji związku M:N są identyczne zarówno dla związków jednostronnie obowiązkowych, jak i obustronnie opcjonalnych. Reguły te są następujące.**

1. Związek M:N jest reprezentowany w modelu relacyjnym poprzez dodatkową relację.

2. Nazwa relacji reprezentującej związek M:N jest złączeniem nazw relacji powstałych z encji połączonych tym związkiem.

3. Relacja dodatkowa zawiera klucze obce wskazujące na klucze podstawowe relacji powstałych z powiązanych encji.

4. Ograniczenia referencyjne są definiowane dla kluczy obcych.

5. Klucze obce tworzą klucz podstawowy relacji. W konsekwencji, ich wartości nigdy nie będą puste.