1. část - Datový model (ERD) a model případů užití

- Datový model (ER diagram) zachycující strukturu dat, resp. požadavky na data v databázi, vyjádřený v notaci UML diagramu tříd (jako na přednáškách) nebo jako ER diagram např. v tzv. Crow's Foot notaci a model případů užití vyjádřený jako diagram případů užití v notaci UML reprezentující požadavky na poskytovanou funkcionalitu aplikace používající databázi navrženého datového modelu. Datový model musí obsahovat alespoň jeden vztah generalizace/specializace (tedy nějakou entitu/třídu a nějakou její specializovanou entitu/podtřídu spojené vztahem generalizace/specializace; vč. použití správné notace vztahu generalizace/specializace v diagramu).
- Odevzdává se dokument obsahující výše uvedené modely včetně stručného popisu datového modelu. Z popisu musí být zřejmý význam jednotlivých entitních a vztahových množin.

2. část - SQL skript pro vytvoření objektů schématu databáze

- SQL skript vytvářející základní objekty schématu databáze, jako jsou tabulky vč. definice integritních omezení (zejména primárních a cizích klíčů), a naplňující vytvořené tabulky ukázkovými daty. Vytvořené schéma databáze musí odpovídat datovému modelu z předchozí části projektu a musí splňovat požadavky uvedené v následujících bodech (je samozřejmě vhodné opravit chyby a nedostatky, které se v ER diagramu objevily, popř. provést dílčí změny vedoucí ke kvalitnějšímu řešení).
- V tabulkách databázového schématu musí být alespoň jeden sloupec se speciálním omezením hodnot, např. rodné číslo či evidenční číslo pojištěnce (RČ), identifikační číslo osoby/podnikatelského subjektu (IČ), identifikační číslo lékařského pracoviště (IČPE), ISBN či ISSN, číslo bankovního účtu (vizte také tajemství čísla účtu), atp. Databáze musí v tomto sloupci povolit pouze platné hodnoty (implementujte pomocí CHECK integritního omezení).
- V tabulkách databázového schématu musí být vhodná realizace vztahu generalizace/specializace určená pro čistě relační databázi, tedy musí být vhodně převeden uvedený vztah a související entity datového modelu na schéma relační databáze. Zvolený způsob převodu generalizace/specializace do schéma relační databáze musí být stručně vysvětlen (v komentáři SQL kódu).
- Skript také musí obsahovat automatické generování hodnot primárního klíče nějaké tabulky ze sekvence (např. pokud bude při vkládání záznamů do dané tabulky hodnota primárního klíče nedefinována, tj. NULL).

3. část - SQL skript s dotazy SELECT

- SQL skript, který nejprve vytvoří základní objekty schéma databáze a naplní tabulky ukázkovými daty (stejně jako skript v bodě 2) a poté provede několik dotazů SELECT.
- Konkrétně musí tento skript obsahovat alespoň dva dotazy využívající spojení dvou tabulek, jeden využívající spojení tří tabulek, dva dotazy s klauzulí GROUP BY a agregační funkcí, jeden dotaz obsahující predikát EXISTS a jeden dotaz s predikátem IN s vnořeným selectem (nikoliv IN s množinou konstantních dat), tj. celkem minimálně 7 dotazů. U každého z dotazů musí být (v komentáři SQL kódu) popsáno srozumitelně, jaká data hledá daný dotaz (jaká je jeho funkce v aplikaci).

4. část - SQL skript pro vytvoření pokročilých objektů schématu databáze

- SQL skript, který nejprve vytvoří základní objekty schéma databáze a naplní tabulky ukázkovými daty (stejně jako skript v bodě 2), a poté zadefinuje či vytvoří pokročilá omezení či objekty databáze dle upřesňujících požadavků zadání. Dále skript bude obsahovat ukázkové příkazy manipulace dat a dotazy demonstrující použití výše zmiňovaných omezení a objektů tohoto skriptu (např. pro demonstraci použití indexů zavolá nejprve skript EXPLAIN PLAN na dotaz bez indexu, poté vytvoří index, a nakonec zavolá EXPLAIN PLAN na dotaz s indexem; pro demostranci databázového triggeru se provede manipulace s daty, která vyvolá daný trigger; atp.).
- Tento SQL skript musí konkrétně obsahovat vše z následujících
    - vytvoření alespoň dvou netriviálních databázových triggerů vč. jejich předvedení,
    - vytvoření alespoň dvou netriviálních uložených procedur vč. jejich předvedení, ve kterých se musí (dohromady) vyskytovat alespoň jednou kurzor, ošetření výjimek a použití proměnné s datovým typem odkazujícím se na řádek či typ sloupce tabulky (`table_name.column_name%TYPE` nebo `table_name%ROWTYPE`),
    - explicitní vytvoření alespoň jednoho indexu tak, aby pomohl optimalizovat zpracování dotazů, přičemž musí být uveden také příslušný dotaz, na který má index vliv, a na obhajobě vysvětlen způsob využití indexu v tomto dotazu (toto lze zkombinovat s EXPLAIN PLAN, vizte dále),
    - alespoň jedno použití EXPLAIN PLAN pro výpis plánu provedení databazového dotazu se spojením alespoň dvou tabulek, agregační funkcí a klauzulí GROUP BY, přičemž na obhajobě musí být srozumitelně popsáno a vysvětleno, jak proběhne dle toho výpisu plánu provedení dotazu, vč. objasnění použitých prostředků pro jeho urychlení (např. použití indexu, druhu spojení, atp.), a dále musí být navrnut způsob, jak konkrétně by bylo možné dotaz dále urychlit (např. zavedením nového indexu), navržený způsob proveden (např. vytvořen index), zopakován EXPLAIN PLAN a jeho výsledek porovnán s výsledkem před provedením navrženého způsobu urychlení,
    - definici přístupových práv k databázovým objektům pro druhého člena týmu,
    - vytvoření alespoň jednoho materializovaného pohledu patřící druhému členu týmu a používající tabulky definované prvním členem týmu (nutno mít již definována přístupová práva), vč. SQL příkazů/dotazů ukazujících, jak materializovaný pohled funguje,
    - vytvoření jednoho komplexního dotazu SELECT využívajícího klauzuli WITH a operátor CASE. V poznámce musí být uvedeno, jaká data dotaz získává.
- Řešení projektu může volitelně obsahovat také další prvky neuvedené explicitně v předchozích bodech či větší počet nebo složitost prvků uvedených. Takové řešení pak může být považováno za nadstandardní řešení a oceněno dalšími body. Příkladem nadstandardního řešení může být řešení obsahující
    - klientskou aplikaci realizovánou v libovolném programovacím jazyce, přičemž práce s aplikací odpovídá případům užití uvedených v řešení 1. části projektu – tedy aplikace by neměla pouze zobrazovat obecným způsobem tabulky s daty a nabízet možnost vkládání nových či úpravy a mazání původních dát, ale měla by odpovídat pracovním postupům uživatelů (např. knihovník po příchodu čtenáře žádá ID průkazky čtenáře, systém vypíše existující výpůjčky čtenáře s vyznačením případných pokut, knihovník má možnost označit jednolivé výpůjčky jako právě vrácené, případně inkasovat pokuty spojené s výpůjčkami, či přidat nové výpůjčky daného čtenáře),
    - SQL dotazy a příkazy ukazující transakční zpracování, vč. jejich popisu a vysvětlení na obhajobě – např. ukázka atomicity transakcí při souběžném přístupu více uživatelů/spojení k jedněm datům, ukázka zamykání, atp.