

# Semestrální projekt TDS II.

Níže najdete osnovu pro vypracování semestrálního projektu, s tím že musíte mít kusy kódu pro všechny požadované úkoly (pokud to vaše struktura nepodporuje, je třeba to doplnit). Je nutné do projektu zakomponovat všechny příkazy a možnosti, které jsou obsaženy v kurzu OA PL/SQL. Výčet níže je pouze hrubý náčrt toho co by se v projektu mělo objevit (čísla sekcí nemusí odpovídat novému kurzu v on-line prostředí). Studenti, kteří mají zapsáno TDS I by měli nejprve odevzdat projekt z TDS I a ten pak rozšířit o požadavky z TDS II.

K vypracování použijte nástroje Oracle SQL Data Modeler a SQL Developer s tím, že vše musí fungovat na školním serveru Oracle, pro nějž vám zašlu přihlašovací údaje.

- Lekce 1
  - Anonymní procedura – obsahující:
  - Deklaraci
  - Blok
  - Výjimka
- Lekce 2
- V deklaraci doplnit vlastní proměnné:
  - Proměnná typu String, Integer, Datum, BOOLEAN
  - Definovat konstantu
  - Definovat proměnnou s defaultní hodnotou
  - Proměnnou s použitím %TYPE
  - Použití implicitní explicitní konverze datových typů
  - Výpis výsledků na konzolu (DBMS.PUT\_LINE)
  - Napsat vnořenou proceduru (minimálně jedno zanoření)
- Lekce 3
  - Napište proceduru, která bude obsahovat INSERT, UPDATE, DELETE, MERGE
  - Použít v proceduře SELECT atr INTO prom ...
  - Použijte v rámci manipulace s daty SQL%FOUND, SQL%NOTFOUND, SQL%ROWCOUNT
  - Použijte v rámci procedury ošetření transakce, COMMIT, ROLLBACK, SAVEPOINT
- Lekce 4
  - Implementujte proceduru s IF-THEN-ELSE a IF-ELSEIF-ELSE
  - Implementujte proceduru s CASE-WHEN-ELSE (s proměnnou za CASE) jako samostatnou implementaci, nebo jako výsledek přiřazení do proměnné.
  - Implementujte CASE-WHEN-ELSE s podmínkou za WHEN
  - Implementujte proceduru řešící všechny varianty kombinací logických operátorů AND, OR, NOT se vstupními hodnotami TRUE, FALSE, NULL
  - Implementujte cyklus s LOOP – EXIT [WHEN] – END LOOP
  - Implementujte cyklus WHILE – podmínka LOOP – END LOOP
  - Implementujte cyklus FOR počítadlo IN spodní..horní LOOP – END LOOP
  - Implementujte cyklus s reverzním čítačem REVERSE
  - Implementujte zanořené cykly s ukončením EXIT a návěštími
- Lekce 5
  - Implementujte explicitní kurzor v deklaraci CURSOR v těle procedury OPEN-FETCH-CLOSE

- Implementujte kurzor s načtením do záznamu INTO v deklaraci %ROWTYPE V implementaci kurzoru použijte příznaky %ISOPEN, %NOTFOUND, %FOUND, %ROWCOUNT
- Implementujte kurzor FOR název záznamu LOOP název kurzoru END LOOP použijte uvnitř kurzoru také EXIT WHEN
- Implementujte kurzor s parametry, které se budou zadávat v těle procedury CURSOR název (parametry) IS ...
- Implementujte kurzor pro UPDATE s WAIT a NOWAIT
- Implementujte více zanořených kurzorů, vnitřní kurzor je ovlivněn parametrem vnějšího kurzoru
- Lekce 6
  - Implementujte uživatelsky definované záznamy %ROWTYPE převzaté z tabulky
  - Definice vlastního typu záznamu TYPE datový typ IS RECORD (...)
  - Tabulka (pole) INDEXED BY TABLE a INDEXED BY TABLE OF RECORDS, INDEX BY BINARY\_INTEGER
- Lekce 7
  - Ošetřování výjimek EXCEPTION WHEN ... THEN s WHEN OTHERS THEN
  - Zachytávání výjimek Oracle serveru předdefinované (konstanty), nepředdefinované (číslované chyby)
  - Deklarace vlastního názvu výjimky DECLARATION název výjimky EXCEPTION, PRAGMA EXCEPTION INIT ...
  - Výpis chybového stavu a komentáře k výjimce: SQLCODE, SQLERRM
  - Uživatelsky definované výjimky RAISE\_APPLICATION\_ERROR s definovaným názvem a druhá varianta s číslem vlastní výjimky a komentářem přímo v těle procedury
  - Vyvolání výjimky v zanořené proceduře a její ošetření v zanořené nebo v nadřazené
- Lekce 8
  - Vytvoření procedury – i předchozí příkazy můžete tvořit jako zanořené procedury
  - Použití parametrů u procedur IN, OUT, IN OUT
  - Zkuste, co se stane, když vložíte nějakou hodnotu do parametru OUT a bude s ní chtít v těle procedury pracovat
  - Přiřazování parametrů pořadím, nebo přiřazením podle názvu název parametru => hodnota parametru, použití DEFAULT hodnoty u parametru procedury
- Lekce 9
  - Vytvořte funkci s návratovou hodnotou CREATE OR REPLACE FUNCTION název funkce (param,...) RETURN datový typ IS ...
  - Použijte vámi definovanou funkci přímo v příkazu jazyka SQL, v sekce SELECT, WHERE, GROUP BY, ORDER BY
  - Prohlédněte si datový slovník s vašimi zdroji, USER\_TABLES, USER\_INDEXES, USER\_SOURCES, USER\_OBJECTS
  - Napište dvě vzájemně se volající procedury s vyvoláním výjimek
  - Ověřte oprávnění k manipulaci s jednotlivými objekty TABULKY, SEKVENCE, POHLEDY, PROCEDUREY
  - Zkuste přidělování uživatelských práv pro výše uvedené objekty a povolené operace
- Lekce 10
  - Vytvořte PACKAGE, s dříve implementovanými procedurami a funkcemi PACKAGE (hlavičku), PACKAGE BODY (tělo)
  - Přetěžování podprogramů – definujte PACKAGE, kde bude mít jedna procedura (funkce) několik interpretací (provádění), podle počtu a typu vstupních parametrů Implementuje FUNKCI, která bude mít libovolný počet textových parametrů, které následně spojí v opačném pořadí dohromady, nebo vymyslete jinou alternativní funkci (proceduru)

- Lekce 11
  - Vyzkoušejte perzistentní stav proměnných (globálních proměnných) v PACKAGE
  - Použijte PACKAGE podporované (nabízené) ORACLE, pro každý uvedený (ne často používaný) PACKAGE (DBMS\_LOB, DBMS\_LOCK, DBMS\_OUTPUT, http, UTL\_FILE, UTL\_MAIL, DBMS\_SCHEDULER) vyzkoušejte jeho funkce
- Lekce 12
  - Použijte dynamicky generované SQL pomocí PACKAGE DBMS\_SQL Implementujte EXECUTE IMMEDIATE s variantou samostatného kódu, předávání parametrů INTO a definice vstupně výstupních parametrů USING IN OUR param, ...
  - Vyzkoušejte ALTER PROCEDURE ... COMPILE a totéž pro FUNCTION, PACKAGE SPECIFICATION A PACKAGE BODY
  - Implementujte postupy pro zvýšení výkonnosti vašich procedur, ukažte příklad zrychlení vykonávání s použitím a bez použití: NOCOPY, DETERMINISTIC, FORALL, BULK COLLECT, SELECT, FETCH a RETURNING
- Lekce 13
  - Vytvořte několik triggerů ve variantách BEFORE AFTER FOR EACH ROW
  - Vytvořte trigger na složený pohled (složený z více tabulek), vložte nový záznam do pohledu a pomocí INSTEAD OF vložte data do zdrojových tabulek
  - Napište trigger využívající proměnné :NEW a :OLD
  - Napište jediný trigger s větvením pro více databázových operací například: BEFORE INSERT OR UPDATE OR DELETE ON tab ... IF DELETED THEN ...
  - Napište trigger, který bude generovat problém mutujících tabulek (Mutation-error)
- Lekce 14
  - Prozkoumejte závislosti mezi objekty, například vzájemně se volající procedury, pohledy z tabulek atd.
  - Vypište stroj závislosti, například z tabulky, je vytvořen pohled, který je používán v jedné proceduře a ta je následně používána v druhé proceduře
  - Popište možnosti závislostí pomocí TIMESTAMP a SIGNATURE (REMOTE\_DEPENDENCIES\_MODE = SIGNATURE)
- Lekce 15
  - Vyzkoušejte optimalizační parametry Oracle, PLSQL\_CODE\_TYPE, PLSQL\_OPTIMIZATION\_LEVEL, a vypište jejich nastavení z pohledu USER\_PLSQL\_OBJECT\_SETTING
  - Vyzkoušejte různé úrovně optimalizace v závislosti na rychlosti provádění složité procedury
  - Vyzkoušejte výpis varování kompilátoru PSQL\_WARNINGS, DBMS\_WARNINGS
  - Vyzkoušejte podmíněnou kompilaci v závislosti na verzi Oracle SQL Serveru, DBMS\_DB\_VERSION, \$IF \$END
  - Zkuste skrýt pro jednu vámi vybranou proceduru její zdrojový kód DBMS\_DDL.CREATE\_WRAPPED