**ТЕСТОВОЕ ЗАДАНИЕ**

**Привет!**

Рады предложить к выполнению тестовое задание.

На выполнение дается 3 дня. Результат прошу направить на почту: olga.borisova@chelpipegroup.com

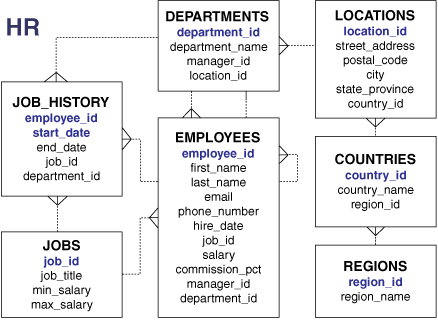
Желаю удачи!

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 

**Условия тестового задания:**

**Цель задания**: разработать пакеты PL/SQL, которые будут обеспечивать интерфейс, скрывающий низкоуровневые функции работы с БД для внешнего приложения.

Для выполнения тестового задания за основу берем стандартную учебную схему Oracle “HR”:



При необходимости, виртуальную машину с тестовой БД можно взять на официальном сайте Oracle <https://www.oracle.com/database/technologies/databaseappdev-vm.html>

Или скачать по ссылке <https://disk.yandex.ru/d/YUvQqXPFEF_GqQ> Пароль к архиву: qaG$mnWi53

Доработка структуры БД

Создать новую таблицу **MESSAGES**, которая будет содержать очередь сообщений для обработки внешней системой. В таблицу добавить колонки:

* ID - первичный ключ, идентификатор сообщения в очереди, должен присваиваться автоматически
* MSG\_TEXT - будет содержать текст сообщения
* MSG\_TYPE – текстовое значение типа сообщения (email, sms и т.п.)
* DEST\_ADDR – адрес получателя сообщения (email, номер телефона)
* MSG\_STATE - числовой статус обработки сообщения внешней системой (0 - добавлено в очередь, 1 - успешно отправлено, -1 - отправлено с ошибкой)

В таблицу **EMPLOYEES** добавить дополнительные колонки:

* UPD\_COUNTER числового типа для счетчика оптимистичной блокировки
* CRT\_USER текстового типа для хранения имени пользователя, создавшего запись в таблице
* CRT\_DATE для хранения даты создания записи
* UPD\_USER текстового типа для хранения имени пользователя, обновившего запись
* UPD\_DATE для хранения даты обновления данных

Триггеры

Для обработки событий DML таблицы **EMPLOYEES** создать триггер (или обновить существующий в учебной схеме HR), который должен добавлять/обновлять указанные выше поля в таблице. В качестве имени пользователя использовать имя текущего пользователя БД.

При необходимости, создать триггер для определения новых значений первичного ключа в таблице **MESSAGES**.

Пакеты

Создать пакет **tabEMPLOYEES** для обработки операций чтения/записи данных в таблицу **EMPLOYEES**

Пакет должен содержать следующие процедуры:

PROCEDURE sel(p\_id IN EMPLOYEES.EMPLOYEE\_ID%TYPE, p\_row OUT EMPLOYEES%ROWTYPE, p\_forUpdate IN BOOLEAN := FALSE, p\_rase IN BOOLEAN := TRUE)

Процедура выполняет извлечение записи по ключу из таблицы EMPLOYEES

Если параметр p\_forUpdate истина, то выполняется SELECT … FOR UPDATE, в противном случае обычный SELECT

При значении истина в параметре p\_rase происходит вызов исключений, в противном случае исключения игнорируются

 PROCEDURE ins(p\_row IN EMPLOYEES%ROWTYPE, p\_update IN BOOLEAN := FALSE)

Выполняет вставку новой строки

При истинном значении параметра p\_update, если строка с таким индексом уже существует, выполняется обновление данных.

Процедура выбрасывает исключения при дублировании строк и нарушении других ограничений, наложенных на таблицу.

PROCEDURE upd(p\_row IN EMPLOYEES%ROWTYPE, p\_insert IN BOOLEAN := FALSE)

Процедура выполняет обновление данных в строке (кроме первичного ключа).

При истинном значении параметра p\_ insert, если строка с таким индексом не существует, выполняется вставка новой строки.

PROCEDURE del(p\_id IN EMPLOYEES.EMPLOYEE\_ID%TYPE)

Процедура выполняет удаление строки данных

FUNCTION exists(p\_id IN EMPLOYEES.EMPLOYEE\_ID%TYPE) RETURN BOOLEAN

Возвращает истину, если строка с указанным ключом существует в таблице

Создать пакет **entEMPLOYEES** для обработки бизнес-логики объектов из таблицы **EMPLOYEES**

Пакет должен содержать следующие процедуры:

PROCEDURE employment(FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, JOB\_ID, DEPARTMENT\_ID, SALARY, COMMISSION\_PCT)

Процедура реализует функционал приема на работу нового сотрудника.

Параметры SALARY, COMMISSION\_PCT не обязательны для заполнения. Если они пустые, при добавлении записи эти данные заполняются средними значениями по подразделению и штатной должности (JOB\_ID, DEPARTMENT\_ID)

Процедура выбрасывает исключения при нарушении ограничений на данные таблицы.

В случае успешного добавления записи в таблицу EMPLOYEES необходимо создать новые сообщения типа email в таблице MESSAGES:

* Для вновь принятого работника: “Уважаемый < FIRST\_NAME > < LAST\_NAME >! Вы приняты в качестве < JOB\_TITLE > в подразделение < DEPARTMENT\_NAME >. Ваш руководитель: < JOB\_TITLE > < FIRST\_NAME > < LAST\_NAME >”. Имя и должность руководителя определить из соответствующих таблиц по DEPARTMENT\_ID и MANAGER\_ID
* Для его руководителя: “Уважаемый < FIRST\_NAME > < LAST\_NAME >! В ваше подразделение принят новый сотрудник < FIRST\_NAME > < LAST\_NAME > в должности < JOB\_TITLE > с окладом < SALARY >”. Значение полей извлечь из соответствующих таблиц по DEPARTMENT\_ID и MANAGER\_ID

PROCEDURE payrise(EMPLOYEE\_ID, SALARY)

Процедура реализует повышение оклада сотруднику

Если SALARY пусто, необходимо повысить оклад на 10%

В случае превышения максимального оклада по должности (MAX\_SALARY) необходимо выбросить исключение

В случае успешного обновления данных в таблице EMPLOYEES создать новое сообщение для руководителя сотрудника следующего вида: “Уважаемый < FIRST\_NAME > < LAST\_NAME >! Вашему сотруднику < FIRST\_NAME > < LAST\_NAME > увеличен оклад с < SALARY old > до < SALARY new >”

PROCEDURE leave(EMPLOYEE\_ID)

Процедура реализует увольнение сотрудника.

Для увольнения необходимо в таблице EMPLOYEES очистить значение поля DEPARTMENT\_ID.

В случае успешного обновления создать сообщение руководителю уволенного сотрудника следующего вида: “Уважаемый < FIRST\_NAME > < LAST\_NAME >! Из вашего подразделения уволен сотрудник < FIRST\_NAME > < LAST\_NAME > с должности < JOB\_TITLE >.”