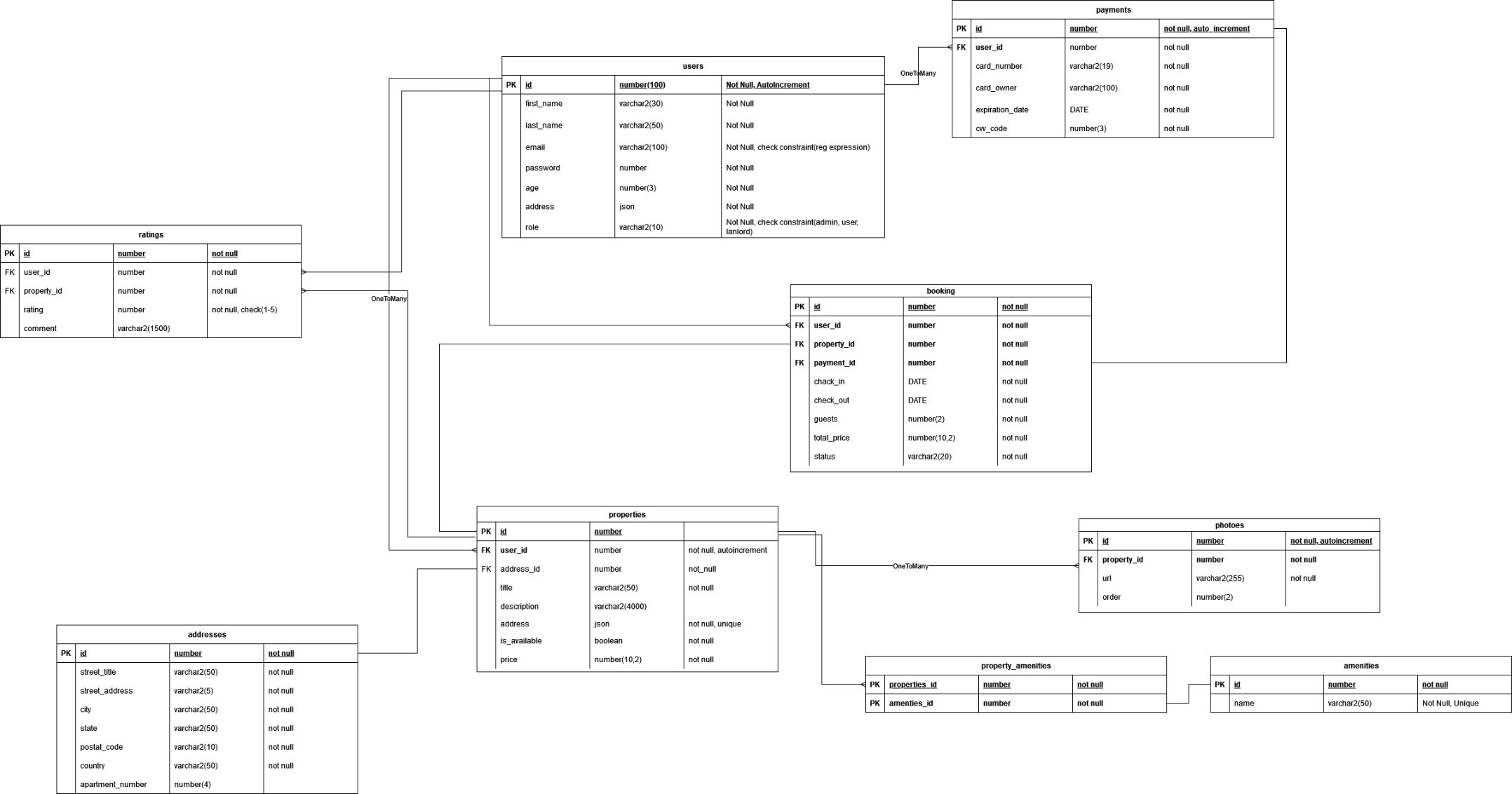
**Приложение 1**

****

Ниже приведён код пакета процедур фильтрации

|  |
| --- |
| CREATE OR REPLACE PACKAGE BODY property\_search  AS  PROCEDURE search\_by\_keyword (  p\_keyword IN VARCHAR2  )  AS  v\_results SYS\_REFCURSOR;  v\_property\_id properties.id%TYPE;  v\_title properties.title%TYPE;  v\_type properties.type%TYPE;  v\_description properties.description%TYPE;  v\_rooms properties.rooms%TYPE;  v\_is\_available properties.is\_available%TYPE;  v\_price properties.price%TYPE;  v\_street\_title addresses.street\_title%TYPE;  v\_street\_address addresses.street\_address%TYPE;  BEGIN  OPEN v\_results FOR  SELECT p.id, p.title, p.type, p.description, p.rooms, p.is\_available, p.price,  a.street\_title, a.street\_address  FROM properties p  JOIN addresses a ON p.address\_id = a.id  WHERE p.title LIKE '%' || p\_keyword || '%'  OR p.description LIKE '%' || p\_keyword || '%';  LOOP  FETCH v\_results INTO v\_property\_id, v\_title, v\_type, v\_description, v\_rooms, v\_is\_available, v\_price,  v\_street\_title, v\_street\_address;  EXIT WHEN v\_results%NOTFOUND;  DBMS\_OUTPUT.PUT\_LINE('----------Поиск по ключевому слову-----------');  DBMS\_OUTPUT.PUT\_LINE('Название: ' || v\_title);  DBMS\_OUTPUT.PUT\_LINE('Тип: ' || v\_type);  DBMS\_OUTPUT.PUT\_LINE('Адрес: ' || v\_street\_title || ', ' || v\_street\_address);  DBMS\_OUTPUT.PUT\_LINE('Описание: ' || v\_description);  DBMS\_OUTPUT.PUT\_LINE('Количество комнат: ' || v\_rooms);  DBMS\_OUTPUT.PUT\_LINE('Доступность: ' || v\_is\_available);  DBMS\_OUTPUT.PUT\_LINE('Цена: ' || v\_price);  DBMS\_OUTPUT.PUT\_LINE('-------------------------------------');  END LOOP;  CLOSE v\_results;  END search\_by\_keyword;  PROCEDURE search\_by\_type (  p\_type IN properties.type%TYPE  )  AS  v\_results SYS\_REFCURSOR;  v\_property\_id properties.id%TYPE;  v\_title properties.title%TYPE;  v\_type properties.type%TYPE;  v\_description properties.description%TYPE;  v\_rooms properties.rooms%TYPE;  v\_is\_available properties.is\_available%TYPE;  v\_price properties.price%TYPE;  v\_street\_title addresses.street\_title%TYPE;  v\_street\_address addresses.street\_address%TYPE;  BEGIN  OPEN v\_results FOR  SELECT p.id, p.title, p.type, p.description, p.rooms, p.is\_available, p.price,  a.street\_title, a.street\_address  FROM properties p  JOIN addresses a ON p.address\_id = a.id  WHERE p.type = p\_type;  LOOP  FETCH v\_results INTO v\_property\_id, v\_title, v\_type, v\_description, v\_rooms, v\_is\_available, v\_price,  v\_street\_title, v\_street\_address;  EXIT WHEN v\_results%NOTFOUND;  DBMS\_OUTPUT.PUT\_LINE('-------------Поиск по типу жилья----------------');  DBMS\_OUTPUT.PUT\_LINE('Название: ' || v\_title);  DBMS\_OUTPUT.PUT\_LINE('Тип: ' || v\_type);  DBMS\_OUTPUT.PUT\_LINE('Адрес: ' || v\_street\_title || ', ' || v\_street\_address);  DBMS\_OUTPUT.PUT\_LINE('Описание: ' || v\_description);  DBMS\_OUTPUT.PUT\_LINE('Количество комнат: ' || v\_rooms);  DBMS\_OUTPUT.PUT\_LINE('Доступность: ' || v\_is\_available);  DBMS\_OUTPUT.PUT\_LINE('Цена: ' || v\_price);  DBMS\_OUTPUT.PUT\_LINE('-------------------------------------');  END LOOP;  CLOSE v\_results;  END search\_by\_type;  PROCEDURE search\_by\_city (  p\_city IN addresses.city%TYPE  )  AS  v\_results SYS\_REFCURSOR;  v\_property\_id properties.id%TYPE;  v\_title properties.title%TYPE;  v\_type properties.type%TYPE;  v\_description properties.description%TYPE;  v\_rooms properties.rooms%TYPE;  v\_is\_available properties.is\_available%TYPE;  v\_price properties.price%TYPE;  v\_street\_title addresses.street\_title%TYPE;  v\_street\_address addresses.street\_address%TYPE;  BEGIN  OPEN v\_results FOR  SELECT p.id, p.title, p.type, p.description, p.rooms, p.is\_available, p.price,  a.street\_title, a.street\_address  FROM properties p  JOIN addresses a ON p.address\_id = a.id  WHERE a.city = p\_city;  LOOP  FETCH v\_results INTO v\_property\_id, v\_title, v\_type, v\_description, v\_rooms, v\_is\_available, v\_price,  v\_street\_title, v\_street\_address;  EXIT WHEN v\_results%NOTFOUND;  DBMS\_OUTPUT.PUT\_LINE('------------Поиск по городу---------------');  DBMS\_OUTPUT.PUT\_LINE('Название: ' || v\_title);  DBMS\_OUTPUT.PUT\_LINE('Тип: ' || v\_type);  DBMS\_OUTPUT.PUT\_LINE('Адрес: ' || v\_street\_title || ', ' || v\_street\_address);  DBMS\_OUTPUT.PUT\_LINE('Описание: ' || v\_description);  DBMS\_OUTPUT.PUT\_LINE('Количество комнат: ' || v\_rooms);  DBMS\_OUTPUT.PUT\_LINE('Доступность: ' || v\_is\_available);  DBMS\_OUTPUT.PUT\_LINE('Цена: ' || v\_price);  DBMS\_OUTPUT.PUT\_LINE('-------------------------------------');  END LOOP;  CLOSE v\_results;  END search\_by\_city;  PROCEDURE search\_by\_price\_range (  p\_min\_price IN properties.price%TYPE,  p\_max\_price IN properties.price%TYPE  )  AS  v\_results SYS\_REFCURSOR;  v\_property\_id properties.id%TYPE;  v\_title properties.title%TYPE;  v\_type properties.type%TYPE;  v\_description properties.description%TYPE;  v\_rooms properties.rooms%TYPE;  v\_is\_available properties.is\_available%TYPE;  v\_price properties.price%TYPE;  v\_street\_title addresses.street\_title%TYPE;  v\_street\_address addresses.street\_address%TYPE;  BEGIN  OPEN v\_results FOR  SELECT p.id, p.title, p.type, p.description, p.rooms, p.is\_available, p.price,  a.street\_title, a.street\_address  FROM properties p  JOIN addresses a ON p.address\_id = a.id  WHERE p.price BETWEEN p\_min\_price AND p\_max\_price;  LOOP  FETCH v\_results INTO v\_property\_id, v\_title, v\_type, v\_description, v\_rooms, v\_is\_available, v\_price,  v\_street\_title, v\_street\_address;  EXIT WHEN v\_results%NOTFOUND;  DBMS\_OUTPUT.PUT\_LINE('----------Поиск по ценовому диапазону------------');  DBMS\_OUTPUT.PUT\_LINE('Название: ' || v\_title);  DBMS\_OUTPUT.PUT\_LINE('Тип: ' || v\_type);  DBMS\_OUTPUT.PUT\_LINE('Адрес: ' || v\_street\_title || ', ' || v\_street\_address);  DBMS\_OUTPUT.PUT\_LINE('Описание: ' || v\_description);  DBMS\_OUTPUT.PUT\_LINE('Количество комнат: ' || v\_rooms);  DBMS\_OUTPUT.PUT\_LINE('Доступность: ' || v\_is\_available);  DBMS\_OUTPUT.PUT\_LINE('Цена: ' || v\_price);  DBMS\_OUTPUT.PUT\_LINE('-------------------------------------');  END LOOP;  CLOSE v\_results;  END search\_by\_price\_range;  PROCEDURE search\_by\_availability (  p\_is\_available IN properties.is\_available%TYPE  )  AS  v\_results SYS\_REFCURSOR;  v\_property\_id properties.id%TYPE;  v\_title properties.title%TYPE;  v\_type properties.type%TYPE;  v\_description properties.description%TYPE;  v\_rooms properties.rooms%TYPE;  v\_is\_available properties.is\_available%TYPE;  v\_price properties.price%TYPE;  v\_street\_title addresses.street\_title%TYPE;  v\_street\_address addresses.street\_address%TYPE;  BEGIN  OPEN v\_results FOR  SELECT p.id, p.title, p.type, p.description, p.rooms, p.is\_available, p.price,  a.street\_title, a.street\_address  FROM properties p  JOIN addresses a ON p.address\_id = a.id  WHERE p.is\_available = p\_is\_available;  LOOP  FETCH v\_results INTO v\_property\_id, v\_title, v\_type, v\_description, v\_rooms, v\_is\_available, v\_price,  v\_street\_title, v\_street\_address;  EXIT WHEN v\_results%NOTFOUND;  DBMS\_OUTPUT.PUT\_LINE('-------------------------------------');  DBMS\_OUTPUT.PUT\_LINE('----------Поиск по доступности-------------');  DBMS\_OUTPUT.PUT\_LINE('Название: ' || v\_title);  DBMS\_OUTPUT.PUT\_LINE('Тип: ' || v\_type);  DBMS\_OUTPUT.PUT\_LINE('Адрес: ' || v\_street\_title || ', ' || v\_street\_address);  DBMS\_OUTPUT.PUT\_LINE('Описание: ' || v\_description);  DBMS\_OUTPUT.PUT\_LINE('Количество комнат: ' || v\_rooms);  DBMS\_OUTPUT.PUT\_LINE('Доступность: ' || v\_is\_available);  DBMS\_OUTPUT.PUT\_LINE('Цена: ' || v\_price);  DBMS\_OUTPUT.PUT\_LINE('-------------------------------------');  END LOOP;  CLOSE v\_results;  END search\_by\_availability;  END property\_search; |

Процедуры экспорта в JSON

|  |
| --- |
| CREATE OR REPLACE PROCEDURE export\_users\_to\_json AS  v\_json CLOB;  v\_chunk\_size CONSTANT INTEGER := 4096;  v\_start\_pos INTEGER := 1;  v\_end\_pos INTEGER;  v\_chunk VARCHAR2(4096);  file\_handle UTL\_FILE.FILE\_TYPE;  BEGIN  -- Создание JSON из данных таблицы users  SELECT JSON\_ARRAYAGG(  JSON\_OBJECT(  'id' VALUE id,  'first\_name' VALUE first\_name,  'last\_name' VALUE last\_name,  'email' VALUE email,  'password' VALUE password, -- В реальных приложениях пароли должны храниться зашифрованными  'age' VALUE age,  'role' VALUE role  )  ) INTO v\_json  FROM users;    -- Вывод JSON в консоль (для проверки)  DBMS\_OUTPUT.PUT\_LINE(v\_json);  -- Запись JSON в файл  BEGIN  DBMS\_OUTPUT.PUT\_LINE('Opening file for appending...');  file\_handle := UTL\_FILE.FOPEN('JSON\_OUTPUT', 'users.json', 'w', 4096);  DBMS\_OUTPUT.PUT\_LINE('File opened successfully.');    -- Цикл записи JSON по частям  LOOP  v\_end\_pos := v\_start\_pos + v\_chunk\_size - 1;  -- Проверка, что не достигли конца JSON  IF v\_end\_pos > LENGTH(v\_json) THEN  v\_end\_pos := LENGTH(v\_json);  END IF;  -- Извлечение части JSON  v\_chunk := SUBSTR(v\_json, v\_start\_pos, v\_end\_pos - v\_start\_pos + 1);  -- Запись части JSON в файл  UTL\_FILE.PUT\_LINE(file\_handle, v\_chunk);  -- Переход к следующей части  v\_start\_pos := v\_end\_pos + 1;  -- Выход из цикла, если достигнут конец JSON  EXIT WHEN v\_start\_pos > LENGTH(v\_json);  END LOOP;  DBMS\_OUTPUT.PUT\_LINE('JSON written to file.');  UTL\_FILE.FCLOSE(file\_handle);  DBMS\_OUTPUT.PUT\_LINE('File closed successfully.');  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('Ошибка записи файла: ' || SQLERRM);  END;  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('Ошибка: ' || SQLERRM);  END export\_users\_to\_json;  CREATE OR REPLACE PROCEDURE export\_payments\_to\_json AS  v\_json CLOB;  v\_chunk\_size CONSTANT INTEGER := 4096;  v\_start\_pos INTEGER := 1;  v\_end\_pos INTEGER;  v\_chunk VARCHAR2(4096);  file\_handle UTL\_FILE.FILE\_TYPE;  BEGIN  -- Создание JSON из данных таблицы payments  SELECT JSON\_ARRAYAGG(  JSON\_OBJECT(  'id' VALUE id,  'user\_id' VALUE user\_id,  'card\_number' VALUE card\_number,  'card\_owner' VALUE card\_owner,  'expiration\_date' VALUE TO\_CHAR(expiration\_date, 'YYYY-MM-DD'),  'cvv\_code' VALUE cvv\_code  )  ) INTO v\_json  FROM payments;    -- Вывод JSON в консоль (для проверки)  DBMS\_OUTPUT.PUT\_LINE('Generated JSON: ' || v\_json);  -- Запись JSON в файл  BEGIN  DBMS\_OUTPUT.PUT\_LINE('Opening file for appending...');  file\_handle := UTL\_FILE.FOPEN('JSON\_OUTPUT', 'payments.json', 'w', 4096);  DBMS\_OUTPUT.PUT\_LINE('File opened successfully.');    -- Цикл записи JSON по частям  LOOP  v\_end\_pos := v\_start\_pos + v\_chunk\_size - 1;  -- Проверка, что не достигли конца JSON  IF v\_end\_pos > LENGTH(v\_json) THEN  v\_end\_pos := LENGTH(v\_json);  END IF;  -- Извлечение части JSON  v\_chunk := SUBSTR(v\_json, v\_start\_pos, v\_end\_pos - v\_start\_pos + 1);  -- Запись части JSON в файл  UTL\_FILE.PUT\_LINE(file\_handle, v\_chunk);  -- Переход к следующей части  v\_start\_pos := v\_end\_pos + 1;  -- Выход из цикла, если достигнут конец JSON  EXIT WHEN v\_start\_pos > LENGTH(v\_json);  END LOOP;  DBMS\_OUTPUT.PUT\_LINE('JSON written to file.');  UTL\_FILE.FCLOSE(file\_handle);  DBMS\_OUTPUT.PUT\_LINE('File closed successfully.');  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('Ошибка записи файла: ' || SQLERRM);  END;  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('Ошибка: ' || SQLERRM);  END export\_payments\_to\_json;  CREATE OR REPLACE PROCEDURE export\_addresses\_to\_json AS  v\_json CLOB;  file\_handle UTL\_FILE.FILE\_TYPE;  BEGIN  -- Создание JSON из данных таблицы payments  SELECT JSON\_ARRAYAGG(  JSON\_OBJECT(  'id' VALUE id,  'street\_title' VALUE street\_title,  'street\_address' VALUE street\_address,  'city' VALUE city,  'state' VALUE state,  'postal\_code' VALUE postal\_code,  'country' VALUE country,  'apartment\_number' VALUE apartment\_number  )  ) INTO v\_json  FROM addresses;    -- Вывод JSON в консоль (для проверки)  DBMS\_OUTPUT.PUT\_LINE('Generated JSON: ' || v\_json);  -- Запись JSON в файл  BEGIN  DBMS\_OUTPUT.PUT\_LINE('Opening file for appending...');  file\_handle := UTL\_FILE.FOPEN('JSON\_OUTPUT', 'addresses.json', 'w', 4096);  DBMS\_OUTPUT.PUT\_LINE('File opened successfully.');    UTL\_FILE.PUT\_RAW(file\_handle, UTL\_RAW.CAST\_TO\_RAW(v\_json));  DBMS\_OUTPUT.PUT\_LINE('JSON written to file.');  UTL\_FILE.FCLOSE(file\_handle);  DBMS\_OUTPUT.PUT\_LINE('File closed successfully.');  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('Ошибка записи файла: ' || SQLERRM);  END;  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('Ошибка: ' || SQLERRM);  END export\_addresses\_to\_json;  CREATE OR REPLACE PROCEDURE export\_properties\_amenities\_to\_json AS  v\_json CLOB;  v\_chunk\_size CONSTANT INTEGER := 4096;  v\_start\_pos INTEGER := 1;  v\_end\_pos INTEGER;  v\_chunk VARCHAR2(4096);  file\_handle UTL\_FILE.FILE\_TYPE;  BEGIN  -- Создание JSON из данных таблицы properties  SELECT JSON\_ARRAYAGG(  JSON\_OBJECT(  'id' VALUE p.id,  'user\_id' VALUE p.user\_id,  'address\_id' VALUE p.address\_id,  'title' VALUE p.title,  'type' VALUE p.type,  'description' VALUE p.description,  'rooms' VALUE p.rooms,  'is\_available' VALUE p.is\_available,  'price' VALUE p.price,  'amenities' VALUE JSON\_ARRAYAGG(a.name)  )  ) INTO v\_json  FROM properties p  LEFT JOIN property\_amenities pa ON p.id = pa.property\_id  LEFT JOIN amenities a ON pa.amenity\_id = a.id  GROUP BY p.id, p.user\_id, p.address\_id, p.title, p.type, p.description, p.rooms, p.is\_available, p.price;    -- Вывод JSON в консоль (для проверки)  DBMS\_OUTPUT.PUT\_LINE('Generated JSON: ' || v\_json);  -- Запись JSON в файл  BEGIN  DBMS\_OUTPUT.PUT\_LINE('Opening file for appending...');  file\_handle := UTL\_FILE.FOPEN('JSON\_OUTPUT', 'properties\_amenities.json', 'w', 4096);  DBMS\_OUTPUT.PUT\_LINE('File opened successfully.');    -- Цикл записи JSON по частям  LOOP  v\_end\_pos := v\_start\_pos + v\_chunk\_size - 1;  -- Проверка, что не достигли конца JSON  IF v\_end\_pos > LENGTH(v\_json) THEN  v\_end\_pos := LENGTH(v\_json);  END IF;  -- Извлечение части JSON  v\_chunk := SUBSTR(v\_json, v\_start\_pos, v\_end\_pos - v\_start\_pos + 1);  -- Запись части JSON в файл  UTL\_FILE.PUT\_LINE(file\_handle, v\_chunk);  -- Переход к следующей части  v\_start\_pos := v\_end\_pos + 1;  -- Выход из цикла, если достигнут конец JSON  EXIT WHEN v\_start\_pos > LENGTH(v\_json);  END LOOP;  DBMS\_OUTPUT.PUT\_LINE('JSON written to file.');  UTL\_FILE.FCLOSE(file\_handle);  DBMS\_OUTPUT.PUT\_LINE('File closed successfully.');  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('Ошибка записи файла: ' || SQLERRM);  END;  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('Ошибка: ' || SQLERRM);  END export\_properties\_amenities\_to\_json;  CREATE OR REPLACE PROCEDURE export\_bookings\_to\_json AS  v\_json CLOB;  v\_chunk\_size CONSTANT INTEGER := 4096;  v\_start\_pos INTEGER := 1;  v\_end\_pos INTEGER;  v\_chunk VARCHAR2(4096);  file\_handle UTL\_FILE.FILE\_TYPE;  BEGIN  -- Создание JSON из данных таблицы bookings  SELECT JSON\_ARRAYAGG(  JSON\_OBJECT(  'id' VALUE b.id,  'user\_id' VALUE b.user\_id,  'property\_id' VALUE b.property\_id,  'payment\_id' VALUE b.payment\_id,  'check\_in' VALUE TO\_CHAR(b.check\_in, 'YYYY-MM-DD'),  'check\_out' VALUE TO\_CHAR(b.check\_out, 'YYYY-MM-DD'),  'guests' VALUE b.guests,  'total\_price' VALUE b.total\_price,  'status' VALUE b.status  )  ) INTO v\_json  FROM bookings b;    -- Вывод JSON в консоль (для проверки)  DBMS\_OUTPUT.PUT\_LINE('Generated JSON: ' || v\_json);  -- Запись JSON в файл  BEGIN  DBMS\_OUTPUT.PUT\_LINE('Opening file for appending...');  file\_handle := UTL\_FILE.FOPEN('JSON\_OUTPUT', 'bookings.json', 'w', 4096);  DBMS\_OUTPUT.PUT\_LINE('File opened successfully.');    -- Цикл записи JSON по частям  LOOP  v\_end\_pos := v\_start\_pos + v\_chunk\_size - 1;  -- Проверка, что не достигли конца JSON  IF v\_end\_pos > LENGTH(v\_json) THEN  v\_end\_pos := LENGTH(v\_json);  END IF;  -- Извлечение части JSON  v\_chunk := SUBSTR(v\_json, v\_start\_pos, v\_end\_pos - v\_start\_pos + 1);  -- Запись части JSON в файл  UTL\_FILE.PUT\_LINE(file\_handle, v\_chunk);  -- Переход к следующей части  v\_start\_pos := v\_end\_pos + 1;  -- Выход из цикла, если достигнут конец JSON  EXIT WHEN v\_start\_pos > LENGTH(v\_json);  END LOOP;  DBMS\_OUTPUT.PUT\_LINE('JSON written to file.');  UTL\_FILE.FCLOSE(file\_handle);  DBMS\_OUTPUT.PUT\_LINE('File closed successfully.');  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('Ошибка записи файла: ' || SQLERRM);  END;  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('Ошибка: ' || SQLERRM);  END export\_bookings\_to\_json;  CREATE OR REPLACE PROCEDURE export\_ratings\_to\_json AS  v\_json CLOB;  v\_chunk\_size CONSTANT INTEGER := 4096;  v\_start\_pos INTEGER := 1;  v\_end\_pos INTEGER;  v\_chunk VARCHAR2(4096);  file\_handle UTL\_FILE.FILE\_TYPE;  BEGIN  -- Создание JSON из данных таблицы ratings  SELECT JSON\_ARRAYAGG(  JSON\_OBJECT(  'id' VALUE r.id,  'user\_id' VALUE r.user\_id,  'property\_id' VALUE r.property\_id,  'rating' VALUE r.rating,  'comment' VALUE r."comment"  )  ) INTO v\_json  FROM ratings r;    -- Вывод JSON в консоль (для проверки)  DBMS\_OUTPUT.PUT\_LINE('Generated JSON: ' || v\_json);  -- Запись JSON в файл  BEGIN  DBMS\_OUTPUT.PUT\_LINE('Opening file for appending...');  file\_handle := UTL\_FILE.FOPEN('JSON\_OUTPUT', 'ratings.json', 'w', 4096);  DBMS\_OUTPUT.PUT\_LINE('File opened successfully.');    -- Цикл записи JSON по частям  LOOP  v\_end\_pos := v\_start\_pos + v\_chunk\_size - 1;  -- Проверка, что не достигли конца JSON  IF v\_end\_pos > LENGTH(v\_json) THEN  v\_end\_pos := LENGTH(v\_json);  END IF;  -- Извлечение части JSON  v\_chunk := SUBSTR(v\_json, v\_start\_pos, v\_end\_pos - v\_start\_pos + 1);  -- Запись части JSON в файл  UTL\_FILE.PUT\_LINE(file\_handle, v\_chunk);  -- Переход к следующей части  v\_start\_pos := v\_end\_pos + 1;  -- Выход из цикла, если достигнут конец JSON  EXIT WHEN v\_start\_pos > LENGTH(v\_json);  END LOOP;  DBMS\_OUTPUT.PUT\_LINE('JSON written to file.');  UTL\_FILE.FCLOSE(file\_handle);  DBMS\_OUTPUT.PUT\_LINE('File closed successfully.');  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('Ошибка записи файла: ' || SQLERRM);  END;  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('Ошибка: ' || SQLERRM);  END export\_ratings\_to\_json;  CREATE OR REPLACE PROCEDURE export\_all\_data\_to\_json AS  BEGIN  -- Вызов процедуры для экспорта пользователей  export\_users\_to\_json;  -- Вызов процедуры для экспорта адресов  export\_addresses\_to\_json;  -- Вызов процедуры для экспорта недвижимости  export\_properties\_amenities\_to\_json;  -- Вызов процедуры для экспорта платежей  export\_payments\_to\_json;  -- Вызов процедуры для экспорта бронирований  export\_bookings\_to\_json;  -- Вызов процедуры для экспорта рейтингов  export\_ratings\_to\_json;  DBMS\_OUTPUT.PUT\_LINE('Экспорт всех данных в JSON завершен.');  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('Ошибка во время экспорта: ' || SQLERRM);  END export\_all\_data\_to\_json; |

Процедуры импорта из JSON

|  |
| --- |
| CREATE OR REPLACE PROCEDURE import\_users\_from\_json(filename IN VARCHAR2) AS  v\_file UTL\_FILE.FILE\_TYPE;  v\_json CLOB;  BEGIN  -- Открываем файл из директории JSON\_IMPORT  v\_file := UTL\_FILE.FOPEN('JSON\_INPUT', filename, 'R');    -- Читаем строку из файла  UTL\_FILE.GET\_LINE(v\_file, v\_json);  -- Вставляем данные в таблицу users  INSERT INTO users (first\_name, last\_name, email, password, age, role)  SELECT jt.first\_name,  jt.last\_name,  jt.email,  jt.password,  jt.age,  jt.role  FROM JSON\_TABLE(v\_json, '$[\*]'  COLUMNS (  first\_name VARCHAR2(30) PATH '$.first\_name',  last\_name VARCHAR2(50) PATH '$.last\_name',  email VARCHAR2(100) PATH '$.email',  password NUMBER(10) PATH '$.password',  age NUMBER(3) PATH '$.age',  role VARCHAR2(10) PATH '$.role'  )  ) jt;  -- Фиксируем транзакцию  COMMIT;  -- Закрываем файл  UTL\_FILE.FCLOSE(v\_file);  EXCEPTION  WHEN OTHERS THEN  IF UTL\_FILE.IS\_OPEN(v\_file) THEN  UTL\_FILE.FCLOSE(v\_file);  END IF;  RAISE;  END;  CREATE OR REPLACE PROCEDURE import\_payments\_from\_json(filename IN VARCHAR2) AS  v\_file UTL\_FILE.FILE\_TYPE;  v\_json CLOB;  BEGIN  -- Открываем файл из директории JSON\_INPUT  v\_file := UTL\_FILE.FOPEN('JSON\_INPUT', filename, 'R');    -- Читаем строку из файла  UTL\_FILE.GET\_LINE(v\_file, v\_json);    -- Вставляем данные в таблицу payments  INSERT INTO payments (user\_id, card\_number, card\_owner, expiration\_date, cvv\_code)  SELECT jt.user\_id,  jt.card\_number,  jt.card\_owner,  TO\_DATE(jt.expiration\_date, 'YYYY-MM-DD'), -- Преобразование строки в дату  jt.cvv\_code  FROM JSON\_TABLE(v\_json, '$[\*]'  COLUMNS (  user\_id NUMBER(20) PATH '$.user\_id',  card\_number VARCHAR2(19) PATH '$.card\_number',  card\_owner VARCHAR2(100) PATH '$.card\_owner',  expiration\_date VARCHAR2(10) PATH '$.expiration\_date', -- Строка в формате YYYY-MM-DD  cvv\_code NUMBER(3) PATH '$.cvv\_code'  )  ) jt;    -- Фиксируем транзакцию  COMMIT;    -- Закрываем файл  UTL\_FILE.FCLOSE(v\_file);  EXCEPTION  WHEN OTHERS THEN  IF UTL\_FILE.IS\_OPEN(v\_file) THEN  UTL\_FILE.FCLOSE(v\_file);  END IF;  RAISE;  END;  CREATE OR REPLACE PROCEDURE IMPORT\_ADDRESSES\_FROM\_JSON(filename IN VARCHAR2) AS  v\_file UTL\_FILE.FILE\_TYPE;  v\_json CLOB;  BEGIN  v\_file := UTL\_FILE.FOPEN('JSON\_INPUT', filename, 'R');  UTL\_FILE.GET\_LINE(v\_file, v\_json);  INSERT INTO addresses (street\_title, street\_address, city, state, postal\_code, country, apartment\_number)  SELECT jt.street\_title,  jt.street\_address,  jt.city,  jt.state,  jt.postal\_code,  jt.country,  jt.apartment\_number  FROM JSON\_TABLE(v\_json, '$[\*]'  COLUMNS (  street\_title VARCHAR2(50) PATH '$.street\_title',  street\_address VARCHAR2(5) PATH '$.street\_address',  city VARCHAR2(50) PATH '$.city',  state VARCHAR2(50) PATH '$.state',  postal\_code VARCHAR2(7) PATH '$.postal\_code',  country VARCHAR2(50) PATH '$.country',  apartment\_number NUMBER(4) PATH '$.apartment\_number'  )  ) jt;  COMMIT;  UTL\_FILE.FCLOSE(v\_file);  EXCEPTION  WHEN OTHERS THEN  IF UTL\_FILE.IS\_OPEN(v\_file) THEN  UTL\_FILE.FCLOSE(v\_file);  END IF;  RAISE;  END; |

Ниже представлены все созданные индексы:

|  |
| --- |
| CREATE INDEX idx\_properties\_type\_city\_price\_available  ON properties (type, address\_id, price, is\_available);  CREATE INDEX idx\_properties\_type ON properties (type);  DROP INDEX idx\_properties\_type;  CREATE INDEX idx\_photoes\_property\_id\_order ON photoes (property\_id, "order");  CREATE INDEX idx\_bookings\_user\_id ON bookings (user\_id);  CREATE INDEX idx\_bookings\_property\_id ON bookings (property\_id);  CREATE INDEX idx\_ratings\_property\_id ON ratings (property\_id);  CREATE INDEX idx\_addresses\_duplicate\_check  ON addresses (street\_title, street\_address, apartment\_number); |