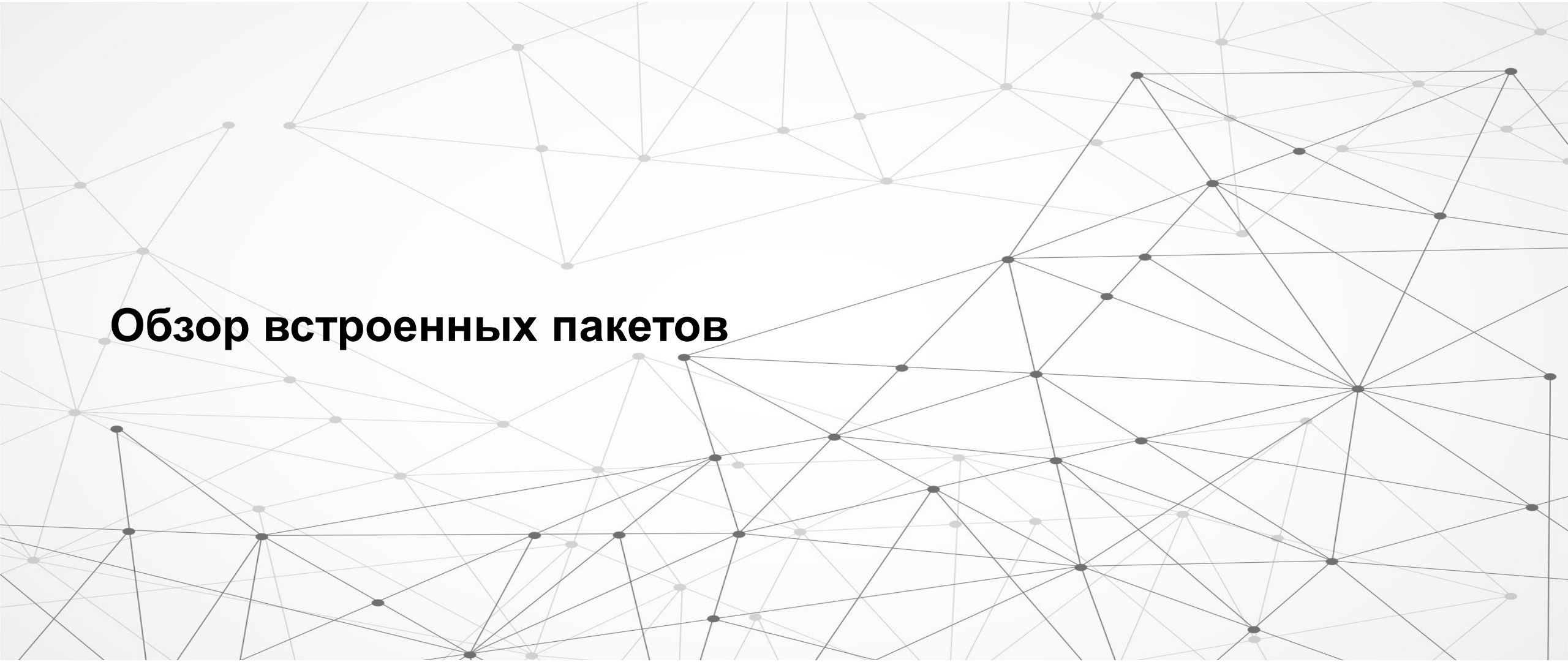
Обзор встроенных пакетов



Data Science Academy

Что изучим

1. Пакеты DBMS_APPLICATION_INFO
2. DBMS_METADATA
3. DBMS_LOB
4. Коротко про поиск запросов (SQL_ID в V$SQLAREA и V$SESSION)

Data Science Academy

Прогресс длительных операций в СУБД Oracle

СУБД Oracle предоставляет несколько средств, которыми можно вопользоваться для этой цели (а также для целей отладки, профилирования, и других, которые подскажет ваша фантазия) - это пакеты

DBMS_APPLICATION_INFO
DBMS_ALERT
DBMS_PIPE

Все эти средства позволяют тем или иным образом организовать передачу данных из одного сеанса работы с СУБД Oracle в другой.

Мы рассмотрим только DBMS_APPLICATION_INFO

Пакет **DBMS_APPLICATION_INFO**

Столбец **module** предназначен для установки имени текущего выполняемого модуля (до 48 байт),
столбец **action** - текущего выполняемого действия (до 32 байт),
и столбец **client_info** - произвольной информации (до 64 байт).
Такие жесткие ограничения на размер данных, связаны с тем, что v$session размещается в оперативной памяти.
Отсюда же высокое быстродействие операций с данной таблицей.

Процедура пакета dbms_application_info	**Устанавливаемые столбцы**
set_module	v$session.module v$session.action
set_action	v$session.action
set_client_info	v_session.client_info
set_session_longops	Столбцы таблицы v$session_longops

Data Science Academy

V$SESSION

Показывает информацию о сеансе связи для каждой текущей сессии в базе данных.
Вы можете запустить запрос к системному представлению, чтобы вернуть всех пользователей, которые в настоящее время имеют работающие соединения в базе данных Oracle/PLSQL.

Синтаксис для получения информации о пользователях, которые вошли в Oracle

```
SELECT USERNAME FROM V$SESSION;
```

Data Science Academy

V$SQLAREA

Показывает статистику по разделяемым областям SQL и содержит одну строку для каждого SQL. Это обеспечивает статистику по SQL операторам, которые находятся в памяти, разобраны, и готовы к выполнению.

```
select * from V$SQLAREA;
```

Data Science Academy

Пакет **DBMS_METADATA**

Часто в работе возникают ситуации , когда необходимо посмотреть исходный код DLL объекта , а специальных графических средств нет под рукой
в этом нам поможет пакет DBMS_METADATA и метод GET_DDL
DBMS_METADATA.GET_DDL -- входные параметры
object_type - Тип объекта (TABLESPACE, CONSTRAINT , CONSTRAINT, Index, package)
name - Наименование объекта например : USERS
schema - схема , по умолчанию схема сессии пользователя или SYS

```
select dbms_metadata.get_ddl(object_type => 'PROCEDURE',name => 'ADD_EMPLOYEE', schema => 'HR') from dual;
```

Data Science Academy

Модуль **DBMS_LOB** используется для работы с четырьмя типами больших объектов (LOB, Large Object):

CLOB,
BLOB,
NCLOB и
BFILE.

Типы данных LOB

SQL Datatype	Описание
BLOB	Двоичный большой объект (Binary Large OBject) Хранит данные в двоичном формата, используется для хранения изображений, аудио и видео, а также компилированного программного кода
CLOB	Символьный большой объект (Character Large Object) Хранит текстовые данные в формате БД
NCLOB	Национальный символьный большой объект (National Character Set Large Object) Хранит текстовые данные в национальной кодировке.
BFILE	Внешний двоичный файл (External Binary File) Файл хранящийся вне базы данных, как файл операционной системы, но доступный из таблиц БД. BFILEs имеет доступ только для чтения. Когда LOB удаляется, Oracle сервер не удаляет сам файл. BFILE поддерживает только случайное(не последовательное) чтение, не участвует в транзакции.

Data Science Academy

Основные правила работы с DBMS_LOB

1. Нельзя использовать в качестве параметра пустой локатор или локатор, имеющий значение null
2. Прежде чем осуществить доступ к внешнему LOBу, ассоциированный с ним файл должен быть открыт
3. Перед завершение pl/sql блока нужно закрывать файл объекта bfile
4. Прежде чем выполнять запись во внутренний большой объект, необходимо заблокировать строку, которая содержит столбец этого объекта.

Вот краткое описание этих функций:
dbms lob.read (локатор, длина, смещение, буфер) – извлекает в буфер PL/SQL указанное количество байтов/символов из идентифицируемого локатором большого объекта, начиная с заданного смещения;
dbms lob.write (локатор, длина, смещение, буфер) – записывает находящееся в буфере PL/SQL указанное количество байтов/символов в идентифицируемый локатором большой объект, начиная с заданного смещения;
dbms lob.append (локатор!, локатор!) – добавляет все содержимое большого объекта, идентифицируемого вторым локатором, в конец большого объекта, идентифицируемого первым локатором;

```
declare
lob CLOB;
textbuf varchar(255);
begin
/* Помещаем в буфер текст, подлежащий вставке */
/* Получаем локатор большого объекта и блокируем
этот объект для обновления */ select document lob into lob from documents where
document >
/* Записываем в объект новые 100 байтов текста */
end;
```

Data Science Academy